



Technical Note

No. 76

Boulder Laboratories

I S O P A R

A NEW AND IMPROVED SYMBOLIC OPTIMIZING
ASSEMBLY ROUTINE FOR THE IBM 650

BY

H. HERBERT HOWE



U. S. DEPARTMENT OF COMMERCE
NATIONAL BUREAU OF STANDARDS

THE NATIONAL BUREAU OF STANDARDS

Functions and Activities

The functions of the National Bureau of Standards are set forth in the Act of Congress, March 3, 1901, as amended by Congress in Public Law 619, 1950. These include the development and maintenance of the national standards of measurement and the provision of means and methods for making measurements consistent with these standards; the determination of physical constants and properties of materials; the development of methods and instruments for testing materials, devices, and structures; advisory services to government agencies on scientific and technical problems; invention and development of devices to serve special needs of the Government; and the development of standard practices, codes, and specifications. The work includes basic and applied research, development, engineering, instrumentation, testing, evaluation, calibration services, and various consultation and information services. Research projects are also performed for other government agencies when the work relates to and supplements the basic program of the Bureau or when the Bureau's unique competence is required. The scope of activities is suggested by the listing of divisions and sections on the inside of the back cover.

Publications

The results of the Bureau's work take the form of either actual equipment and devices or published papers. These papers appear either in the Bureau's own series of publications or in the journals of professional and scientific societies. The Bureau itself publishes three periodicals available from the Government Printing Office: The Journal of Research, published in four separate sections, presents complete scientific and technical papers; the Technical News Bulletin presents summary and preliminary reports on work in progress; and Basic Radio Propagation Predictions provides data for determining the best frequencies to use for radio communications throughout the world. There are also five series of nonperiodical publications: Monographs, Applied Mathematics Series, Handbooks, Miscellaneous Publications, and Technical Notes.

Information on the Bureau's publications can be found in NBS Circular 460, Publications of the National Bureau of Standards (\$1.25) and its Supplement (\$1.50), available from the Superintendent of Documents, Government Printing Office, Washington 25, D.C.

NATIONAL BUREAU OF STANDARDS

Technical Note

76 (PB161577)

July 19, 1960

I S O P A R

A NEW AND IMPROVED SYMBOLIC OPTIMIZING
ASSEMBLY ROUTINE FOR THE IBM 650

by

H. Herbert Howe

NBS Technical Notes are designed to supplement the Bureau's regular publications program. They provide a means for making available scientific data that are of transient or limited interest. Technical Notes may be listed or referred to in the open literature. They are for sale by the Office of Technical Services, U. S. Department of Commerce, Washington 25, D. C.

DISTRIBUTED BY

UNITED STATES DEPARTMENT OF COMMERCE

OFFICE OF TECHNICAL SERVICES

WASHINGTON 25, D. C.

Price \$ 1.50

Contents

Page

1.	Introduction	1
1.1.	What an optimizing assembly routine does.....	1
1.2.	Equipment needed with ISOPAR	1
1.3.	Shortcomings of previous assembly routines	2
1.3.1.	Brevity of remarks.....	2
1.3.2.	Deficient optimization.....	2
2.	Programmer's guide	4
2.1.	Coding form and ISOPAR input	4
2.1.1.	Column 16, card status	5
2.1.2.	Column 17, card type and other information	5
2.1.3.	Columns 24-25, operation.....	6
2.1.4.	Columns 23-25, pseudo-operation	6
2.1.5.	Reservations	7
2.1.5.1.	BLR, Block reservation	7
2.1.5.2.	RBD, Reserve band	7
2.1.6.	Columns 18-22, 26-30, 32-36, addresses	8
2.1.6.1.	Absolute addresses.....	9
2.1.6.2.	Regional addresses.....	9
2.1.6.2.1.	REG, Regional designator.....	9
2.1.6.3.	Blank addresses.....	10
2.1.6.4.	Symbolic addresses.....	11
2.1.7.	Columns 31 and 37, indexing tags.....	11
2.1.8.	Columns 38-75, remarks.....	11
2.2.	Relocatable library subroutines.....	12
2.2.1.	Format of relocatable cards.....	12
2.2.2.	REL, Relocation indicator.....	13
2.2.3.	REQ, Relocatable equivalence.....	13
2.2.4.	RBR, Relocatable block reservation.....	14
2.3.	Other pseudo-operations.....	14
2.3.1.	ALF, Alphabetic word.....	14
2.3.2.	BLA, Block availability.....	14
2.3.3.	HED, Symbol-table clearer.....	15
2.3.4.	SYN, Synonym	15
2.3.5.	EQU, Equivalence.....	16
2.3.6.	BOP, Beginning of program.....	16
2.3.7.	PAT, Punch availability table.....	16
2.4.	800X instructions.....	17
3.	ISOPAR processing instructions	17
3.1.	Assembly on the 650.....	17
3.1.1.	Input card arrangement.....	17
3.1.2.	Operation of the 650 console.....	18
3.1.3.	Programmed stops.....	18
3.1.4.	Other stops.....	18
3.1.5.	ISOPAR output.....	19

	Page
3.2. Reproducing input into output cards_____	20
3.3. Listing the code_____	20
3.3.1. List 1, logical order_____	20
3.3.2. List 2, by D-address_____	21
3.3.3. List 3, by location_____	21
4. Relation of ISOPAR to SOAP II_____	22
4.1. How to use ISOPAR, if already familiar with SOAP II_____	22
4.1.1. Necessary changes in programming methods_____	22
4.1.2. Permissible changes in programming methods_____	23
4.1.3. Some comments in ISOPAR which also apply to SOAP II_____	24
4.2. Improvements in optimization_____	25
4.3. Available drum space for the ISOPAR program_____	26
4.3.1. Use of symbol table_____	27
4.4. Improvements in the 407 control panel_____	28
5. Analysis of the ISOPAR program_____	28
5.1. The optimizing procedure_____	29
5.1.1. Circumstances under which ISOPAR abandons normal processing_____	29
5.1.2. The forward search_____	30
5.1.2.1. Abandon search (QUIT routine)_____	30
5.1.2.2. Initiate backward processing (BACKW routine)_____	30
5.1.2.3. Continue the search_____	31
5.1.2.4. Table of types of addresses_____	31
5.1.3. Some methods of backward processing_____	32
5.1.3.1. Sequence in which addresses are processed_____	33
5.1.3.2. Definition of "fixed address"_____	34
5.1.3.3. Optimizing addends and subtrahends_____	34
5.1.3.4. An example of backward processing_____	35
5.2. Memory allocation_____	36
5.2.1. Input region_____	36
5.2.2. Output region_____	38
5.2.3. Availability table_____	38
5.2.4. Storage region_____	39
5.2.5. Symbol table_____	39
5.2.6. Equivalence table_____	40
5.2.7. Optimizing addends and tags_____	40
5.2.8. Region table_____	41
5.3. Description of the program_____	41
5.3.1. Right-justified temporaries_____	42
5.3.2. Subroutines_____	44
6. Desirable changes in ISOPAR_____	48
6.1. Changes to improve optimization_____	48
6.2. Changes to improve usability_____	49
6.3. Changes to speed assembly and/or make more drum space available_____	50

Illustrations

Page

Figure 1. Samples of ISOPAR input program cards_____	52
Figure 2. Samples of ISOPAR input program cards, continued____	53
Figure 3. Type 533 ISOPAR control panel diagram_____	54
Figure 4. Type 407 ISOPAR control panel diagram_____	55

Tables

Lists of ISOPAR program, by logical order, D-address, and location_____following	55
---	----

ISOPAR, A NEW AND IMPROVED SYMBOLIC OPTIMIZING ASSEMBLY ROUTINE FOR THE IBM 650 *

By

H. Herbert Howe

1. Introduction

1.1. What an Optimizing Assembly Routine Does

The purpose of an assembly routine is to relieve the programmer of some of the problems of assigning storage locations to instructions, data, or "temporaries." When he wishes to refer to a storage location, he writes a symbol, which may be mnemonic, such as SINE; the assembly routine assigns a location to that symbol, and will use the same location whenever that symbol appears again. If it is inconvenient to use a mnemonic symbol, the programmer may select a symbol at random, which is used in the same way.

In the case of the IBM 650, an important feature of any assembly routine is that it should assign locations in such a way as to optimize the program as far as is feasible.

It should be clearly understood that three distinct codes are involved: (1) ISOPAR, which is the code actually being performed. (2) The input code, also called the "symbolic code," which is read and transformed one card or a few cards at a time. (3) The output code, or "object code," which is punched while the input code is being read. The object code is the one which will later be used to make the actual computations which are desired.

1.2. Equipment Needed with ISOPAR

ISOPAR follows its parent SOAP II in that it requires only a basic 650 plus alphabetical device, but it will assemble programs which use other optional features, such as indexing registers, floating point, tapes, printer, disk storage, etc. If the alphabetic device will accept special characters, the programmer has more latitude, but that

*This computer procedure is published as an NBS Technical Note because it may be of general interest and applicability. Its distribution carries no assurance that it is free from error, but only that the author has used it successfully.

feature is not essential. Other special devices are not used in the assembly, even if present. (See 6.3(c).)

1.3. Shortcomings of Previous Assembly Routines

IBM prepared a Symbolic Optimal Assembly Program, commonly known as SOAP, and later a modification called SOAP II. The present assembly routine is modified from SOAP II, but corrects two serious shortcomings in it, as well as several minor ones. On the other hand, ISOPAR still includes a number of known shortcomings, as described in 6. The two serious shortcomings of SOAP II will now be discussed.

1.3.1. Brevity of Remarks

The first shortcoming of SOAP II is largely a matter of control panels, and could have been remedied without much change in the code itself. SOAP II is inadequate in that only ten spaces are allowed for explaining a word of the code; and if the machine includes only the optional Alphabetic feature, but not the Special Character feature, these 10 columns of explanation can include no punctuation. The remarks are often so abbreviated that they suggest meaning only to the person who wrote them, and perhaps not to him after a little time has elapsed. ISOPAR allows 38 card columns for remarks, and they may include any symbols which the 407 will print.

This is accomplished by not trying to carry the remarks from input to output through the 650. Instead, a Reproducer board is permanently wired (see 3.2), and as soon as an ISOPAR code is run through the 650, the cards are run through the Reproducer to reproduce the remarks into the output cards. To simplify coding, this plan has been carried further, and all of the other information on the input cards is transferred to the output by the Reproducer; the only things punched by the 650 are the derived instruction, its location, a card number, and any punches necessary to control the 407 on the basis of defects found during assembly. Sufficient card columns for adequate remarks were obtained by giving up the quite unimportant feature that the output cards should be self-loading.

1.3.2. Deficient Optimization

The second serious shortcoming of SOAP II is one whose remedy required rewriting the entire program, although SOAP II served as a valuable guide throughout. (Since the program was being rewritten, other improvements were included; see 4.) SOAP II produces a program which is rather poorly optimized; the optimization of the output of ISOPAR is much better.

The trouble with SOAP II is that it always processes the cards and addresses in regular sequence: the cards as they reach the hopper, and the addresses in the order L, D, I. The shortcomings of this method of optimization are set forth clearly in IBM's "650 Data Processing System Bulletin: Basic Operation Codes, Program Optimizing, Program Loading," No. G24-5002-0 9/58. On pages 27-28, it is explained that for some operations the time of execution cannot be predicted, and, in order to optimize, one must go ahead to a fixed address and work back. This is just what ISOPAR does.

For example, consider the pair of words:

```
0000 14 ABC
      21 SAVE 0020
```

where 0000 and 0020 are fixed addresses; that is, either absolute values, or regional values, or symbols for which a value has already been assigned when these words are reached. Assuming that its first-choice locations are available, SOAP II would assign values that would make these words read:

```
0000 14 0003 0013
0013 21 0018 0020
```

When this code is performed, the computer wastes 49 word-times between 0018 and 0020.

If operation 14 actually required exactly 10 word-times to perform (or 60, or 110), optimization could not be improved without changing 0000 or 0020. But 14 is not that kind of operation. The time which elapses from the moment it gets the divisor from memory location ABC until it stores the remainder in SAVE is large and variable. In any particular computation, there is a certain dynamic level which ought to be assigned to SAVE to make it fit with ABC; but each time the code is performed, that level will presumably be different. Except for the question of parity (a refinement which need not bother us at this point) any dynamic level is as likely to be suitable for SAVE as any other, so far as fitting it to ABC. (The blank address could be assigned any value, since in any case the second of these words will be in the program register before the division is finished.)

Now consider how these words would be optimized manually. You look ahead until you find a fixed address, and work back. In this instance, working back from 0020, the value 0017 is assigned to SAVE. This will fit ABC as well as anything else and gives the best fit with 0020. It will be seen that, on the average over many performances of the code, the two words will be performed in 49 word-times less than

if SAVE is made equal to 0018: the time interval between ABC and SAVE will be the same, on the average, and the time interval between SAVE and 0020 will always be 49 word-times less.

This is an extreme case. When we consider optimization of many different codes by SOAP II, we see that its method wastes an average of $24 \frac{1}{2}$ word-times per division, as compared with the manual method, if we exclude from consideration those cases in which another variable-length order is reached before we get to a fixed address.

The same argument applies to other variable-length orders, such as multiply, floating operations, read, write. For floating add, the average waste would be somewhat less, because the optimizing addends used in SOAP II are such as will usually waste a few word-times but seldom an entire turn of the drum.

Summarizing, SOAP II wastes considerable time because addresses are always processed in the order in which they are reached. When a variable-length order is reached, it assigns a more or less arbitrary length to it, regardless of what the situation may be on ahead. Since, on the average, it makes little difference in computing time what address is put there, the manual method optimizes the address with relation to later addresses, so that no time is wasted when the next fixed address is reached.

ISOPAR follows the manual method. When a variable-length order is reached, it stops processing and reads in more cards until a fixed address is reached. Then it optimizes backward; then it punches forward. This takes more processing time than SOAP II, but produces a materially better optimized code. In the example cited above, ISOPAR would give, if its first-choice locations were available:

0000	14	0003	1512
1512	21	1517	0020

2. Programmer's Guide

It is assumed that a person attempting symbolic programming, for use with ISOPAR or any other assembly routine, is already at least somewhat familiar with machine-language programming, and no attempt is made to discuss the meanings of the operations or the addresses.

2.1. Coding Form and ISOPAR Input

The ISOPAR input consists of one card for each word of code, with certain additional cards. These additional cards are headings used in listing the code, and cards which serve to guide the assembly. A complete list of the card types is given in 5.2.1.

Figures 1-2 show the coding form used for the ISOPAR input. Each line shows information that might appear in one input card. Numbers at the top show the card columns. It is to be understood that cards would never be used in the sequence and with the entries here used; this is, rather, an assembled collection of different types of cards, given in the order in which they are described in the text.

2.1.1. Column 16, Card Status

Column 16 may be used for a letter which shows the status of the card. Such letters may be omitted if desired, since they do not affect the assembly, and only K affects the 407 listing (by controlling spacing). The following letters are used:

K: The word is a constant, not an instruction. Q: A comparison constant, or the word used to restore a variable instruction after a constant has been subtracted. (Used often with the basic machine; seldom with a machine having index registers.) C: An instruction which is performed immediately after the one preceding it in the listing, and also after some other instruction. N: An instruction which is not performed after the one preceding it in the listing, but is performed after some other instruction. R: An instruction whose logical predecessor is not in the listing at all, but is built up somehow. P: A word used to preset or modify a variable instruction. J: Similar to P, in case the word appears in the code in the sequence in which it will be performed.

These letters are used in the listing of the ISOPAR code. It might be noted, however, that that code has so many interruptions to normal sequence that the value of the letters is rather limited.

2.1.2. Column 17, Card Type and Other Information

Certain of the possible card types are indicated by punches in column 17, and that column also gives some other information about certain cards.

A 1 in 17 indicates a "Comments" card. These are normally heading cards for use in listing, with one at the start of each segment of the code. The comments are written in columns 18-75. The ISOPAR assembly code passes over those cards without using them.

A 2 in 17 indicates a card of a relocatable subroutine. Each actual code-word of the subroutine has such a punch; but the subroutine's type 1 cards, if any, and its REQ and RBR cards do not use 2 in 17, nor does the associated REL card. (See 2.2.)

An X-punch (also known as 11-punch) in 17 indicates a negative word. It is normally used only with constants, including those with 2 in 17. It may be used in an ALF card, but not in a card with 1 in 17.

A Y-punch (also known as 12 or R) in 17 indicates a card of the availability table (2.3.7.). It is not part of the input, but is punched as part of the output, if a PAT card is used.

2.1.3. Columns 24-25, Operation

For every word that is to appear in the object code, there must be a line on the coding form and a card in the input code. The first two digits of the object-code word are entered in columns 24-25 (except in case of an ALF card, 2.3.1.), and are reproduced unchanged into 5-6 of the output card. If the word is an instruction, these digits show the operation; alternatively, they may be the first two digits of a constant. If the word is an instruction, these digits are also used during optimization.

2.1.4. Columns 23-25, Pseudo-Operation

The input symbolic code must also include some "pseudo-operations." A pseudo-operation is indicated by a 3-letter symbol in 23-25.

There are 13 permissible pseudo-operations, listed in 5.2.1. An illegal combination of letters will probably cause the assembly program to go astray. Actually, however, only the lower punches in columns 23 and 25 are used in identifying the pseudo-operations; for example, KAZ or SCI, or 279 would work just the same as BLR. A pseudo-operation is recognized by having blank 17 and non-blank 23.

Pseudo-operation ALF gives rise to a word of code, just as do cards showing actual operations. The other 12 do not. They are used, so to speak, to guide the assembly into correct channels.

Three of the pseudo-operations, BOP, PAT, and HED, are used with only columns 23-25 punched. All of the others require additional punching.

In case RBD cards are used, they must be at the start of the program, preceded perhaps by type 1 cards and a BOP card, but by nothing else. The other pseudo-operations may be used at any time, although normally all except HED also appear before the first card which produces a code-word of the object code.

2.1.5. Reservations

ISOPAR assigns addresses optimally, in cases in which the programmer has not specified the address. When an address is thus assigned, a note is made that that address is now "unavailable," so that it will not be reassigned.

It is always necessary, however, that certain words be in a predetermined relation to each other; e.g., the 10 words of a read band must be together. To make this possible, it is necessary at the start of a program to "reserve" certain portions of the drum, so that the assembly routine will not assign those cells to other uses.

Seven or eight of the pseudo-operations relate to reserving. Two of these are explained here, and the others are explained later.

When a pseudo-operation calls for reserving a location, it is reserved regardless of its previous status; when it is to be "unreserved," that too occurs regardless of previous status.

2.1.5.1. BLR, Block Reservation

Two varieties of BLR cards are illustrated. The first has its I address blank; the second has a four-digit number for I.

The first illustrated BLR card reserves cell 0306, making it unavailable for automatic assignment by the assembly routine. The second illustrated BLR card reserves all cells from 1296 to 1305 inclusive.

2.1.5.2. RBD, Reserve Band

RBD supplements BLR, and reserves large blocks of locations much more rapidly than does BLR.

The first RBD card shown in figure 1 reserves the entire band which starts with 0000, namely locations 0000-0049 inclusive. The second RBD card reserves 0600-1899 inclusive, and also 0000.

The number in the D-address must be a multiple of 50. If it is not, no reservation occurs.

The number in the I-address, if not blank, is normally congruent to 49 (modulo 50). If it is not, reservation extends to the end of the band in which that location lies. Should the I-address be smaller than the D-address, it reserves only one band.

RBD first reserves the initial location of each band, and then reads the next card to see if it is also an RBD card. At the end of the last RBD card, the reservations in the locations $= 0 \pmod{50}$ are extended to the following 49 locations. After that, it also reserves 0000, since that is nearly always desirable. To work correctly, all RBD cards must precede all the rest of the program except possibly type 1 cards and a BOP card.

2.1.6. Columns 18-22, 26-30, 32-36, Addresses

As in all 650 coding, three addresses must be specified for each word: L, the location into which the word is to be placed; D, the data-address in the word; and I, the instruction-address in the word.

In the input code, each of these addresses consists of five characters, and may consist of any combination of letters, digits, blanks, and special characters that is acceptable to the alphabetic input of the 650. Addresses are of four types, as explained later.

It has been found expedient to restrict the generality of the characters in an address. If, for example, an address consists of two consecutive non-blanks and three blanks, it may be hard to tell in a 407 listing which positions are non-blank, and errors have occurred from this source. It is better, therefore, for the programmer to adopt the rule that every address be either left-justified (first position punched) or right-justified (right position punched); or, of course, it may be both left- and right-justified.

The following conventions in writing addresses have been found aids to accurate key-punching. They are illustrated in figures 1-2.

Letters used in addresses are written as capital letters.

An address with fewer than 5 non-blank characters should be written crowded against the left or the right margin of the space on the form, to show left- or right-justification, respectively.

A capital O is written with a bar over it; a zero with a slash through it. The latter may be omitted if the zero is with a group of other digits.

Digit 1 is indicated by a single line; letter I has horizontal bars at the top and bottom.

A blank character within an address is indicated by a small b. Should the programmer desire to have a symbol that is blank at both ends, it should be written with b's at one end to clarify the punching requirements.

2.1.6.1. Absolute Addresses

If the first character of an address is blank and the other four are digits, it is an "absolute address." Such an address is reproduced into the corresponding output card unchanged, unless it is tagged for indexing. If it is a true address, it must of course be a valid one (e.g. for a basic 650, it must be 0000-1999, or 8000-8003). It may also, however, be part of a constant.

ISOPAR reserves an absolute drum address if it is L, since such an address always signifies a location that is being used. (To make the reservation meaningful, the word would have to appear near the start of the code. This is feasible, for example, with constants for which L is absolute.) Absolute D and I addresses are not reserved, (except in relocatable subroutines, 2.2), because such addresses are often parts of constants, or are not true addresses as in operations 80 or 30; and it would reduce available locations to reserve them.

2.1.6.2. Regional Addresses

If the first character of an address is a letter and the other four are digits, it is a "regional address." Regional addresses are used when the programmer knows that a certain group of words must be kept together in a certain sequence (e.g., a table of functions, or the 10 input words), but does not yet want to decide just where to put them. Later, before assembling the program, he will prepare a single REG card to define the region.

Any drum location may be described by a regional address referred to any region. If, for example, the region P is defined by saying that P0001 = 0177, then P0002 = 0178, P0003 = 0179, etc. Going the other direction, that same definition also implies that P0000 = 0176, P9999 = 0175, P9998 = 0174, etc.

2.1.6.2.1. REG, Regional Designator

For each region used in the program, there must be an REG card defining the region. It must appear before the first regional address of that region, and it is normally put near the start of the code.

The two varieties of REG card are shown in figure 1. Although columns 26-30 look like a regional address, the programmer must be careful to realize that they are not.

The first illustrated REG card specifies that the regional address P0001 is equivalent to 0177. It also causes location 0177 to be reserved.

The second REG card specifies that R0001 is equivalent to 0101, and it also reserves all locations from 0101 to 0110 inclusive.

Since it is rare for one to want to use a region consisting of only one word, the first type of REG card is normally used in conjunction with an RBD card. Suppose, for example, region A is to extend from 0800 to 1787, inclusive. One could use a card REG A0800 1787, to specify that A0001 = 0800, and to reserve from 0800 to 1787. This card, however, takes an annoyingly long time to assemble. Assembly may be much hastened by using two cards:

```
RBD  0800 1799
REG  A0800
```

This also reserves 1788-1799, which are not needed; if it will crowd the drum too much to leave them reserved, they can be unreserved with a BIA card.

The four digits in the D ADDR column of an REG card can be any number from 0001 to 9999 inclusive, although many of them would probably not be useful. The number 0000 cannot be used.

2.1.6.3. Blank Addresses

Addresses are normally left blank when the D or I address of one word is to be equal to the L of the next, and the programmer does not care what the address is. More generally, words for which L, D, and I are all given (e.g., constants) may intervene; also, pseudo-operations and type 1 cards may intervene. The following description will help explain the results which ensue when blanks are incorrectly used.

During forward processing, a blank D or I is filled optimally; and if both are blank, I is made equal to D. A blank L is made equal to the D or I that was last processed forward.

During backward processing, blank L is filled optimally, and blank D or I is made equal to the blank L that was last processed backwards.

Although a constant may follow a word with blank I without error in assembly, such an arrangement might impair optimization. If a constant is encountered during a forward search for a fixed address, the search will be abandoned. Usual practice is to put the constants at the end of the segment in which they are encountered, since the type 1 card at the start of the next segment will in any case cause the search to be abandoned.

When an equivalent is assigned to a blank address, the equivalent is reserved.

2.1.6.4. Symbolic Addresses

Any address that is not absolute, regional, or blank is a symbolic address. A symbolic address is assigned a drum equivalent when it is first processed; thereafter, the same equivalent is used, unless it is redefined (see EQU, SYN) or cleared out of the symbol table (see HED).

When an equivalent is assigned to a symbolic address, three things happen: (1) That location is reserved to prevent conflicting use. (2) The symbol is stored in the symbol table. (3) The equivalent of the symbol is stored at a corresponding place in the equivalence table.

By use of SYN or EQU, it is possible to preassign or reassign the equivalent of a symbol.

2.1.7. Columns 31 and 37, Indexing Tags

When a program is being prepared for a 650 equipped with indexing registers, it is not necessary to add the multiple of 2000 or 200 to the absolute address; and for best optimization, you should not do so. A "tag" is put in column 31 or 37 indicating which index register is to be used with a particular address. Permissible tags are 1,2,3, respectively, or A,B,C, respectively, or even J,K,L. The assembly uses the digit-punch only; or, if the column is blank, it uses zero.

The only types of address that could usefully be indexed are absolute or regional. Since indexing is for the purpose of access in turn to locations having a definite relation to each other, there are no circumstances under which it would be useful to index a blank or symbolic address, and ISOPAR ordinarily does not do so, even if a tag is provided in the input code.

ISOPAR decides whether the address is drum (add 2000, 4000, or 6000 for indexing) or core (add 200, 400, or 600) solely on the basis of whether the given address is less or more than $1999\frac{1}{2}$. No check is made on whether the unindexed address might erroneously be in the ranges 2000-8999 or 9060-9999.

In the tagged example shown in figure 1, if region P is as defined by the REG card above it: P0001 = 0177; P0000 = 0176; tag A causes addition of 2000, and the D address of indexed word comes out as 2176.

2.1.8. Columns 38-75, Remarks

Every card may have remarks punched in columns 38-75, explaining the purpose of the card. These remarks are not used in assembly, but show up if the cards are interpreted or listed. The remarks may be as extensive, within these limits, as suits the fancy of the programmer.

The dotted vertical line in the coding form shows where the remarks must end if they are written on an elite typewriter. In figures 1-2, the remarks overrun the line; they are not actually being punched in cards, but the space is used only to describe the card.

2.2. Relocatable Library Subroutines

A relocatable subroutine consists of a number of type-2 cards, and one or more REQ cards. At times, it also includes an RBR card and one or more type-1 cards.

The type-2 cards are written in absolute language, and are usually optimized better than can be done by any optimizing routine. This is warranted, as a rule, because the same subroutine will be used many times.

In all, three pseudo-operations pertain to relocatable subroutines.

The cards for relocatable subroutines are normally put near the beginning of the input deck, preceding all cards for which addresses are assigned optimally. This is necessary so that the locations to which addresses are relocated will not have been assigned by ISOPAR to other purposes before they are reserved during relocation. It would be possible to use a BLR card to reserve the space for the subroutine, but that is an unnecessary complication.

2.2.1. Format of Relocatable Cards

Each card is identified by a 2 in column 17. It may also have an X in 17, if the output word is to be negative.

All addresses must be absolute, and may be either drum or core.

Normally, drum and core addresses would be relocated by different amounts. Any relocatable address less than 2000 is interpreted as drum; any relocatable address exceeding 1999 is interpreted as core.

If an address is not to be relocated (e.g. the D address of operation 30 or 50, or the D or I "address" of a constant), the fact is indicated by putting some non-blank character in the left-hand position of the address. Most commonly, the letter "F" is used, signifying "fixed."

Such an address (see figure 2) resembles in form a regional address, but it is quite different. In fact, F0002 may have three quite different interpretations according to the type of card.

In an ordinary card, it signifies the second word of region F; in an REG card, it signifies that thereafter regional address F0001 = 0002; in a type 2 card, it signifies that the address is 0002 and that it is not to be relocated. In this last use, the first character may be a digit, a letter, or a special character, all of which produce exactly the same result. These three uses of F0002 are shown in figure 2.

A D or I address may be tagged for indexing, as in normal cards. An indexed drum address could not be written as, say, 2030, because it would then be relocated as a core address, not drum.

If L, D, or I is a relocated drum address, that location will be reserved. Hence, it is not necessary to reserve locations for a relocatable subroutine, provided it is assembled before the program starts assigning locations optimally.

Subroutines prepared for SOAP II may be used with ISOPAR by reproducing the cards, provided they meet the conditions mentioned in 4.1.1. In reproducing, columns 41 and 42 are combined into 17, and columns 43-72 are reproduced into 18-47. The resulting subroutine will have some superfluous cards, since in SOAP II it is necessary to reserve specifically all "temporaries."

2.2.2. REL, Relocation Indicator

In this pseudo-operation, columns 27-30 show the amount that is to be added to each drum address, whether L, D, or I: columns 33-36 show the amount to be added to each core address. For example, the first REL card of figure 2 would add 1200 to each drum address, and 0010 to each core address. If either of these fields is blank, it is interpreted as zero, as in the second illustrated REL card. These relocation amounts must be positive.

An REL card applies to all type-2 cards which follow it until another REL card is reached. Normally, the REL card will be placed immediately before the subroutine to which it applies. In any case, every subroutine must be preceded by an REL card.

If a relocated drum address exceeds 1999, or a relocated core address exceeds 9059, it will be left blank in the output. No check is made, however, on whether a relocated core address is less than 9000.

2.2.3. REQ, Relocatable Equivalence

Every relocatable subroutine must include one or more REQ cards; they do not have a 2 in 17. REQ is similar to EQU (2.3.5), but differs from it in two respects: (1) I must be an absolute address; (2) The I-address is modified by adding to it the amount indicated by the last REL card.

For example, the first REL card shown in figure 2 causes all drum locations to be relocated by 1200. If the entry to the subroutine is 0000 in the unrelocated subroutine, it must be 1200 after relocation. This is accomplished by the illustrated REQ card.

Note that the REQ cards are part of the standard subroutine. On the other hand, the REL card is prepared by the person using the subroutine on the particular job. REL must precede the entire subroutine, including the REQ cards.

2.2.4. RBR, Relocatable Block Reservation

This is like BLR, except that the reserved block is relocated by the amount indicated by the preceding REL card. This pseudo-operation was retained from SOAP II, where it had a use because relocated D and I addresses were not reserved. It is probably useless in ISOPAR.

2.3. Other Pseudo-Operations

2.3.1. ALF, Alphabetic Word

This is the only pseudo-operation which gives rise to a word of code. It is used to facilitate entering information which is to be punched alphabetically by the object code.

The quantity whose alphabetic equivalent is to go into the object code is entered under D ADDR. Under any ordinary circumstances, it will be necessary to specify the location in 18-22. An ALF card is ordinarily a constant, and is so designated in column 16.

The first ALF card in figure 2 will cause the number 00 61 87 00 00 to be put into location 0013, and the latter is reserved because it is an absolute L. The second ALF card will cause 00 00 00 00 00 to be put into the location whose symbolic address is PQ, which must already have been defined.

An ALF card may be negative. It is sometimes necessary to make an alphabetic word negative when printing with an on-line 407, the sign being used to control printing. Also, for punched output, it might be desired to have the sign punch in some non-alphabetic column.

2.3.2. BLA, Block Availability

This pseudo-operation causes a block of the drum to be made available, regardless of whether it was previously available or unavailable. Its format is identical with that of BLR.

The usual use of BLA is in connection with RBD. Suppose, for example, it is desired to reserve 0800-1698. This could be done with one card, BLR 0800 1698, but the operation will be much faster if two cards are used, namely

RBD 0800 1699
BLA 1699

Unless the drum is to be tightly packed, one would omit the second card, and let 1699 stay reserved.

A BLA card is not needed at the start of the assembly, since the entire drum is made available during initialization.

2.3.3. HED, Symbol-Table Clearer

This pseudo-operation is used to clear out of the symbol table all of the symbols which are used in just one segment of the program. This serves a triple purpose: (1) It is much easier to use a mnemonic symbol such as LOOP if it can be erased from the symbol table at the start of each segment of code. (2) It reduces assembly time by curtailing the search. (3) In some cases, it may be necessary to erase some symbols in order to have room in the symbol table.

Symbols are divided into two categories: "Long" symbols, which have a non-blank character in the right-hand position; and "short" symbols, which have the right-hand position blank. Whenever a HED card is read, all short symbols are deleted from the symbol table, but long symbols are left unchanged.

It will be noted that a "long" symbol might consist of but one letter or digit, provided it is right-justified.

It is customary to use a HED card immediately after each type 1 card (2.1.2), unless there is a specific reason for not doing so.

2.3.4. SYN, Synonym

This pseudo-operation causes the symbol written in the D ADDR position to be assigned the equivalent of the I ADDR. The latter may be absolute, regional, or symbolic; if regional or symbolic, it must previously have been defined.

Common uses of this pseudo-operation are: (1) When two different symbols are used in writing the code, and it is later decided that they ought to be the same. Use of SYN makes it unnecessary to go through and change one each time it is used. The SYN card must, for this use, appear after one of the symbols has been defined, but before the second

is used. (2) A symbol may be used for mnemonic reasons, but it is necessary to relate it to other addresses; this is done by a SYN card in which I is absolute or regional. (3) Occasionally, as in ISOPAR itself, symbolic temporaries which would otherwise have different values are given the same absolute value to conserve drum space.

If a symbol has been previously defined, it can be redefined with a SYN card, but this feature is seldom useful.

SYN works as follows: The I address is interpreted to give its absolute equivalent. The symbol in the D address is then made equal to that value. The absolute value of the I address is then reserved if (as is usually the case) it is a drum address; if not, the reserving procedure is skipped

The SYN card of figure 2 will cause right-justified X to have the value 0100, and 0100 is then reserved.

2.3.5. EQU, Equivalence

EQU produces exactly the same result as does SYN, which was just described. The reason for having two symbols is to minimize the conflict with SOAP II, where the two symbols worked differently as regards reserving. In practice, it appears that there is never any objection to reserving, although usually the location will already have been reserved. If in some rare case, the location must be left unreserved, the EQU or SYN card should be followed by a BIA card.

For example, the EQU card of figure 2 will cause the symbol ALPHA to be given the value previously determined for a left-justified BETA. This location will then be reserved, if BETA is less than 2000.

2.3.6. BOP, Beginning of Program

This pseudo-operation makes it possible to assemble two or more programs with a single reading of the ISOPAR code. A BOP card is placed between each pair of programs. In reproducing from input to output, note that, as its name implies, the BOP card is to be considered as belonging to the second code.

A BOP card at the start of the first program would be redundant, since the same presetting steps occur initially without a BOP card.

2.3.7. PAT, Punch Availability Table

The availability table, described in 5.2.3, is a 200-word table used by ISOPAR to remember which drum locations are available. The pseudo-operation PAT could be used at any time during assembly to cause

a punch-out of the table, but in practice it is not likely to be wanted except at the end of the assembly.

The output consists of 50 cards, each showing the 4 availability words of one dynamic level. Each card also contains 4 words to facilitate identification of the 4 availability words. The output has Y in column 17 (see 2.1.2), and may be listed on the same 407 control panel as is used for listing the ISOPAR input or output.

For example, the 13th card of the table shows the availability of all locations whose addresses are $\equiv 12 \pmod{50}$. The first 10 columns are punched 00 0012 0462; the next 10 columns show the availability of locations 0012, 0062, 0112, ..., 0462, respectively, where a 1 means "available" and a 0 means "unavailable." Similarly, card columns 21-30 are punched 00 0512 0962, and columns 31-40 show the availability of 0512 (50) 0962. The rest of that level is similarly shown by the rest of the card.

2.4. 800X Instructions

Sometimes a card is inserted in which the location is an address such as 8002. Such a card will not stop the machine, but it is not used in optimizing the program.

3. ISOPAR Processing Instructions

This section discusses the actual technique of processing a symbolic code, once it has been written and punched into cards.

3.1. Assembly on the 650

3.1.1. Input Card Arrangement

ISOPAR deck: This is a 7 per card condensed deck. The cards are identified by serial numbers in columns 4-6, from 001 through 189.

Input deck which is to be assembled: Usually, but not necessarily, all of the pseudo-operations except HED are put at the start. These are followed by standard subroutines, and then the main code. Sometimes the order of the main code may be altered from the logical order, to improve optimization, for either or both of two reasons: If the parts done most frequently appear first, certain temporaries in the most-used part will be assigned locations to fit those parts, instead of to fit other places where they are used less often. Occasionally, such inversion is useful to give the most-used parts first choice of available locations.

Other input programs, if any, provided each of the latter is preceded by a BOP card.
Three blank cards, if desired, to avoid need for using End-of-File key.

3.1.2. Operation of the 650 Console

Storage Entry Switches: 70 1952 9xxx.
Control Switches: STOP RUN RUN PROGRAM STOP STOP, respectively.
COMPUTER RESET
PROGRAM START.

At the end of the assembly, it stops while trying to do operation 70.

If you wish to punch out the availability table, but have not included a PAT card, send control to 0010.

The starting point for performing the code is 0100, but transfer to that point is automatic.

Figure 3 shows the control panel to be used in the 533.

3.1.3. Programmed Stops

ISOPAR includes a number of programmed stops which would indicate defects in analyzing the different problems to be encountered. The only stops which are likely to occur are:

0111: Symbol table full. PROGRAM START will continue assembly, leaving a blank where a symbol has no equivalent. Each time an undefined symbol appears, the machine will stop again, until such time as a HED card clears part of the symbol table.

0222: Drum packed; i.e., no locations are available for blanks or new symbols. PROGRAM START will continue assembly, leaving blank any addresses which cannot be filled.

3.1.4. Other Stops

The most common stop occurs when a blank card is inadvertently put into the input deck. The minimum permissible punching is either a 1 in 17, with the other columns blank; or digits in 24-25, with the other columns blank.

If ISOPAR tries to process a blank card, the computer stops with the two middle digits of the distributor blank. The safest thing to do is to start over.

If an MDF operation (see 5.1.1.(b)) has blank D, the machine stops with 00 0404 9998 in the program register.

3.1.5. ISOPAR Output

The output of the assembly routine consists of the assembled "object code," at the rate of one word per card. The format is as follows:

- 1-4 Location.
- 5-14 Code word; X overpunched in 14 if negative.
- 15 Control X.
- 76 Possibly a number to indicate a defect in the input code which made complete assembly impossible, as follows:
 - 3: An EQU or SYN card with blank or undefined I.
 - 7: L-address missing.
 - 8: D-address missing.
 - 9: I-address missing.

The following are some of the causes of missing addresses: Availability table full (i.e., no address available as equivalent to new symbol or blank address); symbol table is full, and there is no place to store the equivalent of a new symbol (remedy: use more HED cards); an absolute L address with an impossible value (e.g., 3168).

- 77-80 Card number. Only even numbers are used, to permit insertion of corrections.

There is a one-to-one correspondence between input cards and output cards, except for the availability table, if any. There must, therefore, be an output card for each input card that is not a code word, as well as for those that are. To permit the output code to be read into the 650, these extra cards must be readable by it. To do this without damaging the code, it is provided that each such card have 0000 01 0000 0000 in 1-14, with an identifying X in 9.

This puts a Stop order into 0000. It is well-known that it is undesirable to have a word of the code in 0000, as a check in case the program accidentally goes to a zero word. It is necessary to reserve 0000 before assembling the code, so that it will not be used elsewhere in the code; if there is an RBD card, it automatically reserves 0000. If, perchance, the code is permitted to use 0000, the X-9 cards must be sorted out before the code is used.

3.2. Reproducing Input Into Output Cards

It is customary, after the 650 run is completed, to reproduce columns 16-75 from the input cards to the output cards, so that one listing may show both input and output. A permanently wired Reproducer board for this purpose is kept on hand.

After those columns are reproduced, either the input cards or the output cards may be used to correct the symbolic code for later re-assembly.

3.3. Listing the Code

The input or output code maybe listed on a 407 with the control panel illustrated in figure 4. The resulting lists are illustrated by the 1/card lists of the ISOPAR code.

The output is often listed three times; between lists, the cards are sorted and the Alteration Switches are changed. Switch 1 is optional, N for double spaced and T for single spaced.

3.3.1. List 1, Logical Order

This set-up is also used to list the input code.

The cards are listed in logical order; in the output code, they are in sequence by columns 77-80. Alteration switches 2-6 are on N.

A type 1 card, signifying a heading to a new segment of the code, is preceded by a wide space, caused by a program change.

An extra space is produced whenever L of one word is not equal to I of the preceding word, in the symbolic code. An exception is made when both cards have K in 16. Hence, in following an assembled code, you can be sure that the words run in sequence so long as there is no extra space.

Symbolic addresses which are left-justified are printed slightly to the left of those which are not. This makes it easier to determine from a listing which card columns are punched.

On a store operation, both the symbolic and the assembled D-addresses are printed out in special columns, under control of a 2 in column 24. These make it easy to see when something is stored, and where.

Sometimes an emitted legend prints near the right margin, as described in the following paragraphs. BYPASS and BLANK print under control of punches in column 76 (3.1.5), and appear only in the output code. NOTE prints under control of punching in columns 18-36, and appears in the input code, or in the output code if the input code has been reproduced into the output cards. (See 3.2.)

BYPASS means that a SYN or an EQU card could not be processed because its I-address had not been properly defined.

BLANK means that L, D, or I is left blank because of one of many possible defects in the symbolic code.

NOTE signifies any one of several conditions in the symbolic code, which sometimes mean that an error was made in writing the code. Such signals should be investigated. Subject to the exceptions in the next paragraph, this word prints if any one of the following conditions exists: (a) The preceding card had both blank D and blank I. (b) Preceding card had blank I, and this card has non-blank L. (c) Preceding card had non-blank D and I, and this card has blank L.

Even if one of those conditions does exist, NOTE does not print if any one of the following conditions exists: (a) Run-out, and no card is printing. (b) Card has 1 in column 17. (c) Card has blank column 25. (d) Card is punched in column 23. (e) Preceding card was punched in 23. (f) Preceding card had blank 24.

3.3.2. List 2, By D-Address

Sort the cards on 7-10, and remove X-9 cards. Set Alteration Switches 2-6 on T.

The cards are listed in sequence by D-address, with an extra space whenever the D-address changes. This list is used when it is necessary to examine the uses made of a particular temporary location. Some of the cards for which the D-address is not a true address are omitted from the listing; namely, cards with K in 16, and cards with operations 00,30,01,31,05,35,06,36. When a card is so omitted, the spacing, including the extra space for change in 7-10, is handled as if the card did not exist.

3.3.3. List 3, By Location

For this list, sort the cards on 1-4. Set Alteration Switches 2-4 on T, and Switches 5-6 on N.

This lists all cards, in sequence by location, uniformly single or double spaced. It is used for studying what was initially in each location. By comparing lists 2 and 3, it is possible to study all uses made of any location, except instances where it is reached by modifying an address (with index registers, or otherwise).

4. Relation of ISOPAR to SOAP II

4.1. How to Use ISOPAR, If Already Familiar with SOAP II

In most respects, coding for ISOPAR is done exactly as for SOAP II, and a SOAP II symbolic code could be reproduced into different columns for use with ISOPAR. The exceptions fall into two categories: First, changes in coding procedure which must be made in order for the symbolic code to work; second, changes which are permissible, and knowledge of which will facilitate coding.

To reduce programming conflicts with SOAP II, it is provided that any pseudo-operation used in SOAP II will have meaning, although in seven cases the meaning is somewhat different from the SOAP II meaning. In addition, ISOPAR uses one new pseudo-operation.

4.1.1. Necessary Changes in Programming Methods

A person familiar with SOAP II code-writing must take the following differences into account to write an ISOPAR code:

- (a) All operations must be numeric.
- (b) Symbolic addresses are not allowed in type 2 cards (relocatable subroutines). They are, of course, used in REQ cards.
- (c) Minus signs must be X-punches. (SOAP II permits any punch.) In ISOPAR, the sign appears in the same card column as does the card type.
- (d) EQU and SYN produce identical effects: They reserve the location in question if it is a drum location, but do not try to reserve otherwise.
- (e) In type 2 cards, any non-blank character as the first character of an address makes the address fixed. An address is identified as core or drum solely by its magnitude before relocation.
- (f) Although HED has a quite different meaning, a SOAP II code with HED cards will work with ISOPAR unless a symbol headed in one segment is referred to in another segment.

(g) On a long code, it will probably be essential to use HED cards and "short" symbols, lest the symbol table overflow. (With SOAP II, HED serves only to prevent duplication of symbols.)

(h) Location 0000 must be reserved, unless the X-9 cards are to be sorted out before the code is used. This is no hardship, because 0000 should never be used in the code anyway. (If RBD is used with ISOPAR, it automatically reserves 0000.)

(i) A card such as REG A0000, where 0001 of the region is to be location 0000, is inadmissible.

(j) ISOPAR does not now provide for loading an availability table back onto the drum. This feature was in the program for two years, but no use for it was found. (With SOAP II, some users make up availability tables and load them initially as a quick method of reserving large blocks of memory; but with ISOPAR, pseudo-operation RBD accomplishes the same thing more conveniently and expeditiously.)

4.1.2. Permissible Changes in Programming Methods

In addition to the changes listed in the preceding section, those familiar with SOAP II should note the following differences between SOAP II and ISOPAR, some of which will materially simplify programming.

(a) A new pseudo-operation, RBD, greatly shortens the time required to reserve large blocks of memory. It is explained in 2.1.5.2.

(b) BLR, REG, RBR, or BIA may have blank I-address, if only one location is to be reserved or unreserved.

(c) SYN may have a non-drum address in its I-position. With SOAP II a non-drum I causes trouble; with ISOPAR, it merely omits reserving. In ISOPAR, EQU and SYN are identical.

(d) The function of HED has been changed. In ISOPAR, it clears out of the symbol table all symbols which are not right-justified. HED cards must be used in a long code, lest the symbol table become full; they should be used rather frequently, for the reasons given in 2.3.3. Be sure that the right-hand character in any symbolic address is blank, unless the symbol is actually used in more than one segment of the code.

(e) Comments and remarks may be longer than with SOAP II. They may include any characters which the 407 will print, even those not acceptable to the alphabetic input of the 650.

(f) An address may consist of any combination of 5 characters acceptable to the alphabetic input of the 650, as explained in 2.1.6. The following examples illustrate the differences between SOAP II and ISOPAR in this respect:

<u>Address</u>	<u>SOAP II Interpretation</u>	<u>ISOPAR Interpretation</u>
bbbbbb	Blank	Blank
COOOL	Regional	Regional
COOOB	Symbolic	Symbolic
b1234	Absolute	Absolute
b123b	Stops machine	Symbolic
bABCD	Absolute, namely, 1234	Symbolic
bABCT	Stops machine	Symbolic
bbbb1	Stops machine	Symbolic
lbbbb	Symbolic	Symbolic

(g) There is no need in ISOPAR to write out cards having L in the 800X series, and such cards do not affect optimization.

(h) An absolute L address is reserved. If constants or other words with absolute locations are put at the start of the code, it is unnecessary to reserve their locations separately.

(i) When a type 2 card is processed, any relocated address is reserved, whether L, D, or I. (SOAP II reserves only L.) This change probably renders RBR obsolete.

4.1.3. Some Comments on ISOPAR Which Also Apply to SOAP II

There are a few places in which ISOPAR follows SOAP II, but the SOAP II manual is obscure or even incorrect. The following remarks apply to both of these assembly routines.

A word with blank D-address or I-address is normally followed by a word with blank L, and the assembly routines put the same absolute address into the two places. It is not essential, however, that the words be consecutive: They may, without error, be separated by type 1 cards, by pseudo-operations, or by code words in which L, D, and I are all non-blank. For ISOPAR, however, better optimization may occur if constants are put at the end of a segment rather than immediately after the references to them, because a constant encountered during a forward search will initiate the QUITT routine (see 5.1).

On pseudo-operations BLR, BIA, RBR, and SYN, a location is reserved or unreserved regardless of its previous status.

With an REG card, two distinct steps are involved. E.g., with REG A0175 0225, first 0175 is recorded as the equivalent of the regional address A0001, without reference to the size of the region. Then 0175 through 0225 are reserved, just as if it were a BLR card. This makes it clear why regional addresses have significance whether or not they are within the region originally reserved by the REG card.

A regional address may be made negative, by using complements; e.g., A9936 is an address 65 smaller than A0001.

The meaning of a regional address can be changed. This is done in the SOAP II manual: on page 87, REG D0923 0923 specifies that D0001 = 0923. On page 89, D0953 0953 means that thereafter D0001 will equal 0953 instead. It is necessary that the proper reservations be made initially, as is done on page 65.

The first character of a regional address must be a letter. An address whose first character is a digit or a special character (with a 650 designed to read special characters) is interpreted as symbolic.

An REG card in which the first character of the D-address is blank, numerical, or special character, will not define a region; but it will, nevertheless, reserve the specified interval. E.g., the two cards

BLR	0030	0070
REG	0030	0070

do exactly the same thing; namely, reserve from 0030 to 0070, inclusive. BLR could be abolished entirely; but to do so would not shorten ISOPAR by even one word.

4.2. Improvements in Optimization

There are a number of features in which ISOPAR optimizes a program better than does SOAP II:

(a) The main improvement is in discontinuing forward processing when a variable-length instruction is reached, and searching for a fixed address from which to optimize backward.

(b) A defect in optimization of indexed operations has been remedied. E.g., in SOAP II the symbolic word

0200	69	ABC	0031 A
------	----	-----	--------

is assembled with ABC = 0003, if the first choice is available. Since, however, one word-time is spent in interpreting the indexed I-address,

before it starts to perform the instruction, ABC should be 0004 (mod 50), and that is the value which ISOPAR supplies for it. This saves 49 word-times in the performance of such very rare instructions.

(c) When an add or subtract instruction has a D-address of 8001, the time required is, on the average, one word-time less because the word is already in the distributor. ISOPAR optimizes to utilize this saving.

(d) An indexing operation (50, 51, 80, 81, etc.) takes an extra word-time when a complement cycle is required. SOAP II allows an extra word-time under all conditions; ISOPAR saves it for those cases for which the complement cycle never occurs: I.e., operations 80, 82, 88, when the D-address is 0000-1999.

4.3. Available Drum Space for ISOPAR Program

SOAP II uses 1964 locations. The remaining 36 were not enough to meet the requirements of the extra steps involved in ISOPAR.

A considerable number of locations were made available by the innovation mentioned in 3.2: transferring input to output by Reproducer, not through the 650. This saves not only the transferring instructions, but also other words from the fact that input and output did not have to be packed so tightly.

Additional words were saved by more efficient writing of the code. For example: subroutines 4,5,6,8,9 of SOAP II were combined into one subroutine. A more efficient method of getting into subroutine 7 was used. The exits from various subroutines were combined into subroutine 10.

To make additional space, two features of SOAP II were dropped:

(a) ISOPAR does not provide for symbolic addresses in relocated cards (type 2 cards). This is probably not a serious loss, although it may on rare occasions cause inconvenience.

(b) ISOPAR does not provide for symbolic operations. Since some users may regard this as a serious loss, the point will be argued briefly. Any user of the 650 must learn the numerical codes anyway, if only to interpret the signal lights on the machine, or the results of a tracing routine. The three-letter codes used in SOAP II are much less mnemonic than the variable-length letter codes originally proposed by IBM; yet the user must learn them exactly, and not err by using the wrong letters. The result is that the user must learn exactly two arbitrary names for each operation, namely, the numeric and the symbolic.

By using only numerical operation codes, the problem of learning to code is actually simpler than if one must learn both systems. This change saved considerably more than the 100 words omitted from the symbolic-operation table.

All of these changes, however, did not make enough drum space available. Consideration was given to reducing the symbol table from 400 to 300 words. It was found, however, that 300 would often be too small; for example, ISOPAR itself uses more than 300 symbols.

In order to make the necessary drum space available, a fundamental change was made in the method of using the symbol table, with a concomittant change in the operation of a HED card. These changes, explained in the next section, made it feasible to cut the symbol table still more, and it now contains only 210 locations. ISOPAR has been used successfully for two years, and no instance of insufficient space in the symbol table has occurred.

4.3.1. Use of the Symbol Table

In SOAP II, the spot in the symbol table where the code first looks for a given symbol is a function of the 10-digit value of the symbol, the function being so chosen that changing any one character of the symbol will give a different location. The search for that symbol starts at that point, and proceeds upward. Ordinarily, within a short distance, either the symbol is found; or a blank place is found, which then receives the symbol. If the top of the symbol table is reached, the search continues at the bottom; if it then gets back to the starting point, the symbol table is full.

One obvious fact about symbolic addresses is that most of them are used only within a short range of the program. Only a few are used at widely separated places. In recognition of this fact, SOAP II uses the pseudo-op HED, which puts a specified character into the fifth place of any symbol whose fifth place is blank; hence, different segments of the code may be written without concern for duplication of "short" symbols, provided the different segments are "headed" differently. A symbol that is needed in more than one segment has its fifth character non-blank.

In order to obtain more space for the ISOPAR code, provision is made for clearing the symbol table from time to time of symbols that are used in only one part of the code; in this way, the symbol table can be made shorter. But when this is done, the SOAP II method of storing symbols cannot be used. Suppose, for example, the "function" of a symbol is 1756. Suppose when the symbol is stored, 1756 and 1757 are occupied; it is stored in 1758. Suppose that later 1757 is cleared out. The next time this symbol comes up, the code would find a blank at 1757, and conclude that this symbol was not in the table.

Hence, ISOPAR needed a quite different method, and the following was adopted: As in SOAP II, symbols used in only one segment of the code should be "short" symbols; i.e., with their last position blank. Symbols used in more than one segment should be "long"; i.e., with the last position non-blank. In SOAP II, a short symbol is "Headed" by a HED card; in ISOPAR, all of the short symbols are cleared out of the symbol table by a HED card.

Long symbols are stored in the symbol table starting with the first cell of the table and working upwards. Short symbols are stored starting with the top and working downwards. When a symbol is sought, the code starts at the appropriate end, and checks each cell in turn until the symbol or zero is found. To guard against the possibility that the entire table become full, one cell is left vacant at each end; when the search ends in a zero, a check is made to determine that it is in the middle, and not beyond either end.

By taking care to use long symbols only when necessary, and by clearing out the short symbols at rather frequent intervals, the search-time can be kept within reasonable bounds, although it will usually be somewhat greater than with SOAP II.

4.4. Improvements in the 407 Control Panel

Although not a part of the code itself, the 407 control panel used for listing a code is also important. The ISOPAR 407 control panel is described in 3.3. Here, we mention briefly those features which are improvements over the panel shown in the SOAP II manual for that program: (a) Wide space above a type 1 card. (b) Extra space when L does not equal I of previous card. (c) D-addresses of store operations printed in separate columns. (d) Alignment-shift between addresses which are left-justified and those which are not. (e) The legend "NOTE" is printed on input code as well as on output. (With SOAP II, the legend calls attention to errors only after assembly is finished.) (f) Alteration switches provide for modifications of spacing for three different lists, and for suppression of some cards on List 2.

5. Analysis of the ISOPAR Program

Throughout this analysis, when digits of a word are mentioned, the digits are counted from left to right: The left-most digit is digit 1, and the right-most digit is digit 10. This differs from the method used in the 650 manual and on the console; but it agrees with the method used in many other IBM publications, and is much more natural than counting from the right.

5.1. The Optimizing Procedure

ISOPAR normally assembles a code by considering the addresses in the sequence in which they appear. Usually, L is known from some previous word; D is determined at an optimum distance from L; and I is determined at an optimum distance from either L or D, depending upon the operation. Then it goes on to the next word.

If, however, circumstances are such that it is impossible to optimize an address in this way, ISOPAR starts a "forward search" to find some fixed address from which it can process backwards to the point at which the search started. During the search, enough information must be stored about each card to permit reconstructing it.

The forward search may be terminated in either of two ways: (1) It may prove impossible to find a fixed address; then ISOPAR returns to where it started the search, assigns an address more or less arbitrarily, processes the stored-up cards forward, and resumes normal processing. This is the QUITT routine. (2) It may find a fixed address, whereupon the stored-up cards are processed backwards and the results stored; then these results are punched forwards. This is the BACKW routine. Thereafter, normal processing is resumed.

The different circumstances involved in these processes will now be given in some detail.

5.1.1. Circumstances Under Which ISOPAR Abandons Normal Processing

There are three conditions which cause ISOPAR to stop the normal processing and start a forward search. At the time it starts the search, it makes a note as to which condition initiated the search, because there are slight differences in the later assembly methods.

(a) When L is a symbolic address that has not been previously defined. This is the start of a wholly new series of instructions. Any assignment of this L at this stage would be wholly arbitrary. Hence, it is best to look ahead for a fixed address, and work back.

(b) When a variable-length operation (called MDF operation), such as multiply, divide, float, read, write, table look-up, is encountered. With most MDF operations, it does not matter whether the abandoning occurs after D, or after L of the next word, since I of the MDF operation may have any address whatever. However, operation 84 is different. Since L for that word is used after the table-look-up is completed, the machine cannot start looking for the I-address until after the operation is completed. Hence, for an 84 operation, normal processing should be abandoned and the forward search started between the D and I addresses. To simplify coding, the same procedure is used for all MDF operations.

(c) When a D address is indexed (for use on a 650 which has indexing registers). Since an indexed D address refers to a variable drum location, it is impossible to optimize the I of that word with respect to it. Hence, it is better to go ahead to a fixed address and work back to I. There is no need to start a forward search on indexed I, because the following L will surely be either a fixed address or a new symbol.

5.1.2. The Forward Search

When a card is examined during the search, there are three possible outcomes, which will now be discussed in detail.

5.1.2.1. Abandon Search (QUITTT routine)

If it is impossible to find a fixed address to which to tie the stored-up cards, ISOPAR abandons the search, returns to the point at which the search started, assigns an initial address more or less arbitrarily, and processes the stored-up cards forward. The circumstances under which it does this are:

(i) Storage region for storing cards is full. The region holds 15 cards.

(ii) Another MDF operation is reached.

(iii) D is indexed; since its time of performance is variable, it is impossible to optimize across it.

(iv) I is indexed, meaning that the next instruction is variable.

(v) Card is of type 1 or 2, or is a pseudo-operation. Normally this is the end of a block, and will be followed by fixed L. (See 6.1 (e).)

(vi) An address is of the type designated as QUITTT in 5.1.2.4.

5.1.2.2. Initiate Backward Processing (BACKW Routine)

Should a fixed address be found before any of the conditions listed in 5.1.2.1 occur, the search is successfully completed, and the code starts processing backward. Addresses considered "fixed" for this purpose are indicated in 5.1.2.4 by BACKW.

5.1.2.3. Continue the Search

If nothing occurs to terminate the search, it continues to the next card. In other words, the search continues if none of the special conditions listed in 5.1.2.1 occurs, and if the address is marked ON in 5.1.2.4.

5.1.2.4. Table of Types of Addresses

The following table shows the different types of addresses which may be encountered during the forward search, and shows, for each type, whether the address initiates the QUITT routine or the BACKW routine, or whether the search goes on. These decisions are subject to the rules in 5.1.2.1, which prescribe other conditions under which the search is abandoned.

<u>Line</u>	<u>Type of Address</u>	<u>L</u>	<u>D*</u>	<u>I</u>
1	Blank	ON	ON	ON
2	Regional, region defined	QUITT	BACKW	BACKW
3	Regional, region not defined	QUITT	QUITT	QUITT
4	Symbolic, symbol defined before search started	QUITT	BACKW	BACKW
5	Symbolic, not defined, but room in symbol table	ON	ON	ON
6	Symbolic, not defined, symbol table full	QUITT	QUITT	QUITT
7	Absolute, 0000-1999	QUITT	BACKW	BACKW
8	Absolute, 8000-8003, 8005-8007 (called 300X)	QUITT	ON	QUITT
9	Absolute, 9000-9059	ON	ON	ON
10	Absolute, other	QUITT	QUITT	QUITT

*D is examined only if it is an address, excluding, for example, operation 30. Specifically, it is examined only if the first digit of the optimizing addend (5.2.7) is non-zero.

In the following explanation of the items in this table, an item is identified by its line number and column heading:

For lines 1 and 5, the address is wholly undefined, and the search continues. For 8D and line 9, the address is defined, but its dynamic level is not; therefore, the search continues for an address of fixed dynamic level.

When a fixed D or I is reached (2D, 2I, 4D, 4I, 7D, 7I), the search for a fixed address is successfully terminated, and we begin backward processing.

A fixed L (2L, 4L, 7L) may be reached by the search only if the previous D and I were unfixed. This means the start of a new block of words; the previous block cannot be tied to anything, and we might as well abandon the search.

If we reach an address that cannot be processed (lines 3, 6), the job will have to be done over anyway, and the easiest way of finishing it is best.

For 8I, the location of the next order to be performed is variable, and there is little point in trying to optimize through it. Therefore, the search is abandoned.

8L would indicate a mistake. If there were a word with 800X in L, it should follow one with 800X in I, and the latter terminates the search. (ISOPAR does not encourage the use of cards with 800X in L, and makes no use of them.)

10L is a programmer's mistake, since L cannot have such a value.

10D and 10I are normally parts of a constant. ISOPAR does not attempt to optimize backward through a constant; rather, we QUITT the search upon reaching one.

5L is really two cases. If the undefined symbol is different from any symbol used up to this point, it could just as well read QUITT. More commonly, however, an undefined-symbol L would be a symbol already encountered during the forward search, but not yet put into the symbol table because it has not been processed. We might have, for example:

0047	19	0200	
	45		Y
	46		X
	20	T1	X
X	20	0100	

where 0100 is an address that has been fixed at the time operation 19 initiates a forward search. In this case, to QUITT and process forward from 0200 would assign to X a value that would be unsuitable with reference to 0100, and it is best to pass through the X, and then process backward from 0100. To simplify coding, it is provided that when L is an undefined symbol, the forward search always continues, irrespective of whether that symbol may have been encountered during the search. The only circumstance under which this might detract from the optimization would be if the storage area filled up before a fixed address was reached, but would not fill up had we started a new search with the symbolic L in question.

5.1.3. Some Methods of Backward Processing

During forward processing ISOPAR uses the same procedure as SOAP II: L is determined from some previous word; the optimum dynamic level of D is derived by measuring forward from that of L, and an available

location at or above that dynamic level is found; the dynamic level of I is measured from that of L or of D, according to the operation, and an available location for I is found.

The procedure is more complicated during backward processing, since there are several different cases. The program first examines the D and I addresses to determine whether they are fixed; the method of processing depends upon those two decisions, and also upon the type of operation. In explaining this procedure, we first discuss the methods of handling different types of addresses, and then consider the question of what constitutes a "fixed" address.

5.1.3.1. Sequence in Which Addresses are Processed

The operations encountered during backward processing (since MDF operations are excluded) fall into three categories:

(a) Those in which D is not an address to be optimized (including all shift and indexing operations).

(b) Those in which D is measured from L, and I is measured from D. These are considered under the name "arithmetic operations."

(c) Those in which D and I are both measured from L. These will be considered under the name "branch operations."

The first backward card (the one which initiates backward processing) and the last one (the one which initiated the forward search) require special treatment, and will not be discussed here. For the intermediate cards, the sequence of processing the addresses is as follows:

Shift or indexing operation: D forward, I backward, L backward.
Arithmetic operation, fixed I: I backward, D backward, L backward.
Arithmetic, fixed D, unfixed I: D backward, I forward from D, L backward from D.
Arithmetic, D and I unfixed (An abnormal case, but could occur if D and I are new symbols referring to some point later in the program): I backward, giving an arbitrary address, D backward, L backward.
Branch, D and I unfixed (abnormal case): I backward, L backward, D forward.
Branch, D unfixed, I fixed: I backward, L backward, D forward.
Branch, D fixed, I unfixed: D backward, L backward, I forward.
Branch, D and I fixed: Compute L backward from D and also from I, and then use whichever dynamic level is lower, where

"lower" must be interpreted with reference to a circular drum. This assures that if it branches equally often to D and to I, the time required will be a minimum.

5.1.3.2. Definition of "Fixed Address"

The definition of "fixed address" during backward processing is not the same as during the forward search.

After we have processed a blank L backwards, any blank D or I must be considered as fixed; for it will be the same as the L address of the following word, which will have been fixed by the time the given D or I is reached.

Core I is fixed under the same conditions, for an analogous reason. Its dynamic level is the same as that of the following L, which has already been determined. A core D address, with branch operation, is fixed because it is normally associated with the L of the following word if that is also core. A core D address with arithmetic operation is unfixed, for it will work equally well at any dynamic level. It would be possible to improve the optimization of segments of code involving core addresses by a considerable expansion of the code to take a larger number of conditions into account. (See 6.1 (f).)

The full schedule of fixed-unfixed addresses may be found by studying the one per card code under "Backwards Routine," lines 696-734.

5.1.3.3. Optimizing Addends and Subtrahends

Words 0300-0899, shown in the one per card listing of the program (lines 110-303), and described in 5.2.7, give the addends required for forward optimization. Two addends are given, which are often equal, and never differ by more than one. The first addend is used if the augend is even; the second if the augend is odd. The following examples illustrate how they work:

<u>Augend</u>	<u>Addends</u>	<u>Result</u>
0000	54	0005
0001	54	0005
0002	54	0007
0003	54	0007
0000	43	0004
0001	43	0004
0002	43	0006
0003	43	0006

Now look at this table from the point of view of backward processing. For the case where the addends are 54, if we start back from 0005 we could go to either 0000 or 0001; but since optimization now requires that the result be as large as possible, we take 0001. If we start with 0007, the result is 0003. If we start with 0006, we see that 0002 is too large, since going forward from 0002 gets us to 0007; therefore, going back from 0006 we must take 0001. I.e., in this case, the first part of the table becomes:

<u>Minuend</u>	<u>Result</u>	<u>Subtrahends</u>
0007	0003	54
0006	0001	54
0005	0001	54

where the subtrahends were determined to fit the result to the minuend; the first subtrahend is used if the minuend is even, and the second if it is odd.

Similarly, from the second part of the addend table, we get:

<u>Minuend</u>	<u>Result</u>	<u>Subtrahends</u>
0006	0003	34
0005	0001	34
0004	0001	34

Thus, when the addends for forward processing are 54, the subtrahends for backward processing are also 54; but when the addends are 43, the subtrahends are 34. By considering all cases, it is found that if the larger of the addends is odd, they are used unchanged as subtrahends; if the larger is even, they must be interchanged. For use on an automatic computer, the formula is: If the two addends be called ab , compute the remainder of $(a + b + 1)/4$. If this remainder is zero, the subtrahends are ba ; otherwise, they are ab .

5.1.3.4. An Example of Backward Processing

Consider, now the example given in 5.1.2.4. In the last word, D is fixed. Since the operation is 20, L is measured backward from D , and I is measured forward from D . Assuming throughout that ISOPAR's first choice is available, $X = 1545$, and the last $I = 0003$. The five words now read, inserting values already derived:

0047	19	0200	
	45		Y
	46		1545
	20	T1	1545
1545	20	0100	0003

In word 4, I is now fixed. D is measured backward from I and L is measured backward from D. Deriving optimum values, and inserting them:

0047	19	0200	
	45		Y
	46	1537	1545
1537	20	1542	1545
1545	20	0100	0003

Both D and I of word 3 are now fixed. By the last rule of 5.1.3.1, we find that L measured from D would be 1534, and measured from I it would be 1541. Using the lesser value we now have:

0047	19	0200	
	45	1534	Y
1534	46	1537	1545
1537	20	1542	1545
1545	20	0100	0003

Finally, word 2 is a branch operation with D fixed. Measure L backward from D, and I forward from L:

0047	19	0200	1531
1531	45	1534	0035
1534	46	1537	1545
1537	20	1542	1545
1545	20	0100	0003

5.2. Memory Allocation

The one per card listing of the ISOPAR code shows, near the beginning, the locations of various blocks of memory assigned to particular purposes. Some of these are discussed in the following sections.

5.2.1. Input Region

The following list shows the read-in schedule. The absolute addresses are 0151-0160. Words 1-3 are alphabetic; the others are not. When a card column is shown as 0, it means that a zero is emitted into that position of the word.

<u>Regional Address</u>	<u>Card Columns, and Explanation</u>	
R0001	18-22.	L, alphabetic.
R0002	26-30.	D, alphabetic.
R0003	32-36	I, alphabetic.
R0004	Emitted 00000 00008 if X-17 (negative word). Emitted 00000 00000 otherwise.	
R0005	0 31 0 0 24 25 37 0 0 0. Indexing tags, and operation. If 31 or 37 is blank, it is replaced by an emitted 0.	
R0006	Unused.	
R0007	0 0 0 0 0 0 19 20 21 22.	L, numeric, often invalid.
R0008	0 0 0 0 0 0 27 28 29 30.	D, numeric, often invalid.
R0009	0 0 0 0 0 0 33 34 35 36.	I, numeric, often invalid.
R0010	0 0 0 0 0 0 0 0, followed by the 2-digit card type.	

Shortly after the card is read, control goes to R0010, whereupon the card type becomes the entry to a particular segment of the routine. The 533 control panel determines this type as follows: If column 17 contains a 1 or 2, that digit preceded by 0 is the card type. If 17 lacks 1 and 2, but 23 is punched, then the digit punches (1-9) of 23 and 25 are the card type. If neither of these conditions exists, then 08 is used. The full list of card types is as follows:

- 01 1 in c.c. 17; i.e., comments card.
- 02 2 in c.c. 17; i.e., relocatable subroutine.
- 08 Ordinary card of the code.
- 16 ALF, Alphabetic word.
- 21 BLA, Block availability.
- 25 SYN, Synonym.
- 27 BOP, Beginning of program.
- 29 BLR, Block reservation.
- 54 EQU, Equality.
- 73 PAT, Punch availability table.
- 84 HED, Symbol-table clearer.
- 93 REL, Relocation indicator.
- 94 RBD, Reserve band.
- 97 REG, Regional designator.
- 98 REQ, Relocatable equivalence.
- 99 RBR, Relocatable block reservation.

5.2.2. Output Region

For most output cards, the output region is used as follows:

P0001-4 Not used for punching.
P0005 Card number. Punches into 77-80.
P0006 Location of the assembled instruction. Punches into 1-4.
P0007 OP, in positions 5-6. Punches into 5-6..
P0008 D, at right end of word. Punches into 7-10.
P0009 I, at right end of word. Punches into 11-14.
P0010 Control information, as follows:

Position 1: A card that is not part of the code; i.e., type 1, or a pseudo-operation other than ALF. Causes 0000 01 0000 0000 to punch into 1-14, with an X in 9.
Position 3: Causes a 3 to punch in column 76. (See 3.1.5.)
Position 5: Activates Punch B. (See below)
Position 7: Causes it to omit punching L, and to punch 7 in 76.
Position 8: Causes it to omit punching D, and to punch 8 in 76.
Position 9: Causes it to omit punching I, and to punch 9 in 76.
Position 10: Word is negative. Causes X to overpunch in 14.

All of the cards have a control X punched in 15.

When an ALF card is processed, words 7,8,9 may hold other digits in addition to those described above, but they are not punched and do not affect the result in any other way.

When Punch B is activated (8 in position 5 of word 10), the availability table (2.3.7) is punched. Ten columns are punched, respectively, from each of words 1,2,3,4,9,6,7,8. (Word 5 is not used, to make it possible to carry the card number through such a punch-out.)

5.2.3. Availability Table

The availability table "remembers" drum locations used by the program which is being assembled, and those still available for use. The table contains 200 words, or 2000 digits; each digit shows the status of one memory word: 0 = unavailable; 1 = available. The 10 digits of the first word of the table show the status of, respectively, 0000, 0050, 0100, ..., 0450. The second word shows the status of 0500, 0550, ..., 0950. The fifth word of the table shows the status of 0001, 0051, 0101, ..., 0451, etc. With this arrangement, when we have determined the dynamic level which ought to be used for a blank address or a new symbol, we can tell whether any such address is available by zero-testing four words. If those four words are all zeros, we continue in the proper direction and use the nearest word that is available. In any case, when a non-zero availability word is found, the first non-zero digit of the word is used.

Suppose we need a location of dynamic level 12. If we are processing forward, the choice goes, in sequence, 0012, 0062, 0112,... 1962, 0013, 0063,... 1963, 0014, etc. But if we are processing backward, the sequence of choice is more complicated: 1512, 1562,... 1912, 1962, 1012, 1062,... 1462, 0512, 0562,..., 0962, 0012, 0062,..., 0462, 1511, 1561,..., 1961, 1061, etc. It is often possible to tell just by glancing at an address whether it was derived during backward or forward processing.

5.2.4. Storage Region

When a card is stored during the forward search, five of the input words are stored for later reference. Words R0001 through R0005 (described in 5.2.1) for the card which initiated the search are stored in S0001-S0005; the same words for the next card are stored in S0006-S0010; etc. Words R0006-R0010 need not be stored, because they are not used in processing a type 08 card, which is the only type that is stored up.

If the search ends with transfer to the QUITT routine, S0001-S0005 are brought back to R0001-R0005, and processed. Then each other card is brought in turn to the same place. When the card which initiated QUITT is reached, its words R0006-R0010 are still in place, never having been disturbed.

If the search ends with the BACKW routine, the last card stored is taken to R0001-R0005 for processing backwards; its assembled word is in P0006-P0010 as usual, but these words are then stored at the end of the storage region. Each card is processed in turn going backwards, and stored in the storage region; when all have been processed, they are brought back in turn to P0006-P0010, and the cards are punched at top punching speed.

5.2.5. Symbol Table

When a numerical address has been assigned to a symbol, the 10-digit representation of the symbol is stored in the symbol table. Symbols in which the last character is non-blank are stored as encountered, starting with the beginning of the table. Symbols in which the last character is blank are stored as encountered, starting with the end of the table. These latter are cleared from the table whenever there is a pseudo-operation HED. Care is taken that the two series of symbols do not overlap. The symbol table is cleared during initialization.

5.2.6. Equivalence Table

Whenever a symbol is stored in the symbol table, its numerical equivalent is stored in the equivalence table. The equivalent of the first symbol in the symbol table is stored in the D-position of the first word of the equivalence table; the equivalent of the second symbol goes into the I-position of the same word. The equivalents of the next two symbols are similarly stored in the second equivalence word.

The equivalence table need not be cleared initially or by a HED card, because it is never consulted except to store or find the equivalent of a symbol that is in the symbol table. Storing is done in such a way that only the four digits are disturbed, leaving the rest of the digits and the sign unchanged. The values are used in such a way that the other digits and the sign do not affect what we take from the table.

5.2.7. Optimizing Addends and Tags

Information needed to optimize the instructions is stored in 0800-0899. For example, the information for operation 17 is in location 0817. The digits of one of these words have the following meanings:

- 1: Optimum dynamic level of D minus given level of L, when L is even.
- 2: Same, when L is odd.
- 3-4: Optimum dynamic level of I, measured from given level of L or of D according to the operation, when latter is even.
- 5-6: Same, when reference address is odd.
- 7: 8 for an operation whose execution requires a variable length of time (MDF operation).
9 for an operation requiring a fixed length of time.
- 8-9: 88, operation 31.
98, other shift operations; namely, 30, 35, 36.
89, indexing operations; namely, 50-53, 58-59, 80-83, 88-89.
99, any other operation.
- 10: 8 if I is measured from L (e.g., branch, shift, indexing).
9 if I is measured from D (e.g., add, subtract, store).

The following represent slight modifications in the above:

(a) When digit 1 is zero, D is not an address; it may have other meaning, as in shift operations; or it may be meaningless, as in operations 00, 01.

(b) For shift operations, where the optimizing addends come from analysis of D, the first 6 digits of 0830 etc. are zero. The optimizing addends are given in 1250-1259.

(c) For indexing operations, D is not a drum address. For these operations, digits 5-6 are zeros, but digits 1-4 have a special meaning. They have the values 0101 for operations 80,82,88, and 0000 for the other indexing operations.

(d) For an MDF operation, optimizing addends cannot be given reliably; but digits 1-6 are the quantities that are used in case the forward search ends in QUITT.

5.2.8. Region Table

Locations G0001-G0029 (excepting G0010, G0020, G0021) are used for interpreting regional addresses. Suppose an REG card shows X1369, meaning that thereafter regional address X0001 = 1369. The numerical equivalent of X is 87, and 87-60=27. The number 1369 is then stored in G0027.

The region table is cleared during initialization; a non-zero in any cell of the table shows that there has been an REG card for that region. It will be noted that a card such as REG X0000 is invalid, since it would be impossible to tell that there had been an REG card for that region.

G0012 and G0020 are assigned by SYN cards to particular temporaries. G0021 is used as a zero constant.

5.3. Description of the Program

The main routine is quite short — in fact, it may be said to consist of only 4 words. From the last of these, it branches to any of 16 different exits, namely the card types given in 5.2.1.

On a card type 01, the branch is directly to Subroutine 10. On type 02 or a pseudo-operation, the branch is to a short segment of routine which appears near the end of the list.

On an ordinary card (type 08), the branch is to the segment which immediately follows the main routine. Thence, in turn, the code may branch to any of several subroutines, or to the forward search. The latter uses various subroutines, and eventually goes to either the QUITT routine or to the backward processing (BACKW) routine. It will be noted that under one set of circumstances it goes to BACKW and almost immediately resumes the forward search. This is the case where the search starts with an MDF operation, but I of that word is fixed.

There are three entries to the forward search, because it is necessary to make note of the reason for the search. The third entry is from Subroutine 19.

5.3.1. Right-Justified Temporaries

A number of quantities are stored in temporaries with absolute addresses, or with 5-letter symbolic addresses, because they are used in various parts of the program. Most of these will now be briefly described; for some, the derivation of the 5-letter symbol will be indicated.

ALOPT (all optimizing tags) is developed in Subroutine 14, and is sometimes modified elsewhere. It is developed for each word that is processed. Its digits have the following significance:

- 1: 9, Subroutine 11 is to reserve any location which it selects. This is the initial and usual value for this digit.
- 8, Subroutine 11 is not to reserve any location. This value is used when both D and I are fixed during backward processing of a branch instruction. (5.1.3.1.)
- 2: 8 is initial value. Changed to 9 during backward processing, if it is found that D is not a fixed address. (5.1.3.2.)
- 3: Same, for I.
- 4-5: Unused.
- 6: 8, the operation is one for which D is not an address, or at least not a drum address. (E.g., 30, 80.)
 9, the operation is one for which D is an address. (E.g., 10, 69.)
 The determination is made on the basis of whether the first digit of OPTIM is zero.
- 7-10: Identical with digits 7-10 of OPTIM.

BLANB holds the equivalent found for a blank L during backward processing.

BLANK holds the equivalent of a blank D or I during forward processing.

CDIFF holds the amount by which an REL card says to relocate core addresses.

COUNT shows the number of addresses in an instruction which are tagged for indexing. It is either 0, 1, or 2.

DDIFF holds the amount by which an REL card says to relocate a drum address.

DRUMT is initially 80 0000 0000. It is changed to 90 0000 0000 if the availability table shows that there is no space on the drum to which a blank or symbolic address can be assigned. It is changed to 80 0000 0000 during processing of a BLA card.

EQUIV holds, at its right end, the absolute equivalent of any address that has been processed, as soon as the equivalent has been determined.

HSYMB holds the alphabetic address that is being worked on, excepting that if the address is blank, it is not stored in HSYMB. (The symbol was carried over from SOAP II, where it stood for "headed symbol.")

LSYMB (location of symbol) shows either (1) the location in the symbol table where the symbol given in HSYMB may be found, or (2) an unused location, into which that symbol may be stored. LSYMB is in the D-position of the word.

OPTIM shows the word taken from 0800-0899 (see 5.2.7) that is appropriate to the operation involved in the card now being worked on.

OPREG (operating register) shows the dynamic level of an address which we have just finished using. If it is a drum address, the address itself (perhaps indexed) is put into OPREG, except sometimes for the addition of COUNT on an L-address. If it is a core address or an 800X D address, the dynamic level (never less than -0001 nor more than +0050) is computed and stored in OPREG.

ORCEB, ORCEQ, and SAVOR are used for storing various quantities used in attempting to optimize through core addresses. A core address itself is always optimized.

SAVEL and SAVED are used for saving addresses already derived for L and D, respectively, at the start of a forward search.

1030 and 1050 hold various distributor-branch tags, used to select different cases. The determination of the proper circumstances for changing these tags proved one of the most troublesome features of writing ISOPAR.

1030 holds the QUITT tag. Digit 1 is set to 8 when processing a card of type 08 (ordinary). It is changed to 9 at start of a forward search. It is used in Subroutine 19: A new-symbol L, when processing forward, starts a forward search if this digit is 8; but not if it is 9, for then we are on QUITT routine, and a search has been proven fruitless.

1030 holds other tags, in digits 5 and 6, which are used to tell the circumstances which started the forward search, and which are used at various places in the processing.

1050 holds the First-Card Tag. Its first digit is set to 8 when a card of the main routine is read. It is changed to 9 at the end of the study of a card during the forward search; so if the first card searched initiates QUITT or BACKW, the tag is still 8. During QUITT, it is changed to 8 at the end of the first card; hence, during QUITT, it is 9 for the first stored card if there are more than one; it is 8 in other cases. On BACKW, if there is more than one stored card, 1050 is changed to 8 when a blank L is processed backwards, whereafter it is used to indicate that blank D or I is to be made equal to BLANB. On BACKW with only one stored card, 1050 starts with 8, but changes to 9 if it is determined that the forward search started with L, so that this one card can be processed on the same basis as the first backward card in the more general case.

5.3.2. Subroutines

In addition to the 16 segments of code entered from the 16 different card types, there are 19 subroutines. Although many of these are usually entered from card type 08, some of them can be reached from other portions of the code. The subroutines are numbered 1 through 22, omitting 12, 15, and 16. The salient facts about each subroutine will now be listed, including a statement as to whence it is entered.

The symbolic entry word to a subroutine is of the form SUBR1, SUB19, SB10A, SUB2R, etc., excepting for subroutine 18, the entry word to which is INDEX. At the time of entry, the exit word is in the Distributor unless otherwise noted. Normally, this is stored soon after the subroutine is entered. To conserve drum space, the exits are stored in only three different locations, called EXITX, EXITY, and EXITZ, and these are put into otherwise unused locations of the punch region.

- 1: Initialize the drum at the start of a program. This is entered either from the first word of the main routine, or from BOP. It clears the region and symbol tables; clears card number; sets DRUMT to 80 0000 0000; and makes the availability table all 1's, since the entire drum is initially available.
- 2: Double-entry. If entered at SUB2R, it reserves all locations from FWA to FWA+N, inclusive. If entered at SUB2U, it unreserves similarly. At time of entry, FWA is in Upper, and N is in

Lower. Before reserving is attempted, checks are made so that no reservation will be attempted beyond location 1999. It is entered from Subroutines 4, 8, and 19, and from BLR, BLA, EQU. (Reserving of optimally assigned locations is done by Subroutine 11, without use of Subroutine 2.)

3 is used during the forward search to determine whether an address is fixed. At entry, the address in question, alphabetic, is in upper. At time of exit, type of address is indicated by contents of accumulator, as follows:

Blank, new symbol, core: 00000 00000 00000 00008

Drum, regional, old symbol: 00000 00000 00000 00000

800X: 00000 00000 0800X 00000

Absolute, not in those ranges; undefined symbol with
table full; regional with region undefined:
 00000 00008 00000 00000

4, 5, 6: Used to process L, D, or I, respectively, backwards. This is done principally by appropriate use of other subroutines.

7: This stores a new symbol in the symbol table, and its absolute equivalent in the equivalence table. Makes use of the contents of EQUIV, HSYMB, and LSYMB, which were stored by Subroutines 11, 9, and 9, respectively. Entered from Subroutines 4,5,6,19, 20,21.

8: This is used to process addresses in type 2 cards (relocatable subroutines). At entry time, the Upper contains the alphabetic address, used only to determine whether the address is to be fixed; the Lower contains the numeric address. At exit, the relocated address is in the Lower; or, if the relocation gives too high a value, the Upper will contain 00 0000 8000.

9: This subroutine, which may be entered from Backward Processing or from Subroutines 3, 4, 5, 6, 19, 20, or 21, examines an address to determine its character. At entry, the alphabetic address is in the Upper. Although the basic exit instruction is, as usual, in the Distributor at time of entry, it may be modified by the subroutine according to the type of address. If the I-position at time of entry shows K, the I-position of the actual exit instruction is as follows:

- K-2: Address is regional with region undefined; or is a new symbol but symbol table is full.
- K-1: Address is blank.
- K: Absolute drum address; or symbolic or regional address equivalent to a drum address.
- K+1: Undefined symbol, but there is room for it in symbol table.
- K+2: 800X address.
- K+3: Core address; i.e., absolute, in range 9000-9059.
- K+4: Some other absolute address (or possibly a symbolic or regional address equivalent to an "other" address). Normally, part of a constant.

Subroutine 9 also stores the following:

- HSYMB: The alphabetical symbol, unless it is a blank.
- EQUIV: The absolute equivalent of the address, in cases K, K+2, K+3, and K+4. (Should an undefined symbol happen to consist of four digits preceded by a digit or a special character, the four digits will be stored in EQUIV, but will not be used.)
- LSYMB: If the address is an old symbol, D-position of LSYMB receives the location of the symbol in the symbol table, less the location of the first word of that table. Similarly, if the address is a new symbol, LSYMB receives the location into which the symbol ought to be stored.

10: This so-called subroutine is a collection of short bits of code, namely the common endings of various subroutines and segments.

11: This subroutine is used for finding the optimum address, when the given address is blank or is a new symbol. It uses Subroutine 13 to find the optimum dynamic level; then 11 examines the availability table to find the available location which most nearly fits the prescribed level. The selected location is stored in EQUIV, and also is in the Lower at time of exit. The location is reserved, except when the first digit of ALOPT is 8. (See 5.3.1.)

Subroutine 11 is entered from Subroutines 4,5,6,19,20,21. At entry time, the exit instruction is in the Lower, and includes various branch-distributor tags. If the Lower is of the form 00 0abc KKKK, abc shows where we came from, as follows:

- From Subroutine 4 (backward L): 989
- From Subroutine 5 (backward D): 988
- From Subroutine 6 (backward I): 990
- From Subroutine 19 (forward L): 890
- From Subroutine 20 (forward D): 888
- From Subroutine 21 (forward I): 889

Should the drum be packed, so that no location is available, the exit instruction is changed from 00 0abc KKKK to 00 0abc KKKK+1. Also the drum tag is changed from 80 0000 0000 to 90 0000 0000, so that it will thereafter be unnecessary to go through a search of the availability table.

13: This subroutine is usually entered from Subroutine 11, but possibly from 4, 5, 20, or 21. The data it uses are found in OPREG, OPTIM, and ALOPT, described in 5.3.1. At entry time, the Distributor contains 00 0axc KKKK, where x is irrelevant, KKKK is the location of the next instruction, and ac have the values given under Subroutine 11.

At the end, the computed optimum dynamic level is shown at the right end of the Accumulator, with a value between 00 and +49, inclusive.

To explain the classification of cases within the subroutine, note that we never get to 13 when processing the D address of an indexing or shift operation, or when processing backward D of a branch operation.

14: This is entered from many different places and is used before much else can be done with a word. It transfers the control information and the operation to the output region, and separates the indexing tags and counts them (See COUNT, 5.3.1.). It moves the optimizing addends and tags from 0800-0899 to OPTIM, and develops the initial form of ALOPT. At exit time, ALOPT remains in the Distributor.

17: This subroutine is entered from 5 or 20. At entry, the Lower still contains P, the optimum dynamic level determined by Subroutine 13. 17 computes a slight correction to the dynamic level of D in certain cases: If D is 8002 and P is odd, or if D is 8003 and P is even, the correction is +1; if D is 8001, the correction is -1. This amount will later be added to P in Subroutine 20, or subtracted from P in 5.

18: This modifies a drum or core address if the address was tagged for indexing. At entry, the Lower contains the absolute value of the address to be indexed, in the I position. The D position of the Distributor is 0 if a D-address is being indexed, and 1 if an I-address is being indexed.

19, 20, 21: Used to process L, D, or I, respectively, forward.

22: This is a short subroutine used to preserve ORCEQ at the start of backward processing, for possible use when we go forward again. At time of entry, exit word is in Lower instead of Distributor.

6. Desirable Changes in ISOPAR

ISOPAR was prepared in the Spring of 1958, and has been in constant use since that Fall. It was probably effectively debugged for the features which exist on the 650 used by the National Bureau of Standards. On the other hand, it has never assembled a complete program for a 650 equipped with core, tapes, or disks. (If features involving operations 25-29 are used, the 407 plugboard should be slightly modified.)

Some work has been done toward incorporating certain desirable revisions. However, the National Bureau of Standards will soon stop using the 650, and since the existing ISOPAR represents considerable improvement over SOAP, it seems best to publish ISOPAR now, and indicate possible further improvements.

In preparation for publication, a few minor improvements were made. Most of the changes suggested in the following sections would involve major modifications of the code.

6.1. Changes to Improve Optimization

(a) If the forward search ends with the fixed D address of an MDF operation, it processes backward from that fixed address, but then does not start another forward search; instead, it uses the normal processing procedure from that fixed D. This defect was discovered in debugging the code, but it would require very extensive modification to remedy, and it has not been corrected.

(b) An indexed D with an operation such as 80 should not initiate a forward search, since there is no uncertainty as to the length of time required to perform the operation.

(c) For the same reason, an indexed D with operation 80, encountered during the forward search, should not initiate QUITT, but the search should continue.

(d) When operation 69 has blank D (the usual entrance to a subroutine), the L of the following instruction is correctly made equal to that D; but then a forward search ought to be initiated, since the latter instruction is not performed in the place indicated by its L. This would probably be a major revision: At present, a forward search always starts with either L or I, but this would require one to start with D.

(e) A forward search is terminated by QUITT when the search reaches any type of card other than type 08 (ordinary card); and even by a type 08 card if it is a constant. Present programming custom

often puts together at the end of one segment and the start of another the following cards: Card with blank I, constants, type 01 card, HED card, card with blank L. It would be desirable to carry the search through this group, and optimize backward through them. This would require a substantial increase in the capabilities for storing cards and for processing stored cards.

(f) In ISOPAR, the optimization through core addresses is not very good. In this it resembles SOAP II, although the methods are different. Suppose, for example, that during normal processing the following pair of instructions is reached:

```
1000 60 1003 9002
9003 15 ABC
```

Both assembly routines derive 07 as the dynamic level of 9002, and then use 07 as the dynamic level of 9003 in optimizing ABC. Other cases involve more complicated rules; but essentially they all assume that some nearby core addresses are the same.

Perhaps optimization could be improved by making a record of the dynamic level last assigned to each core address, for use when that address shows up again as L. Or if, as above, a core L is not equal to the preceding I (or D in case of a branch order), a forward search might be initiated on the grounds that we probably have the start of a loop. Such changes would substantially increase the length of the code.

(g) In deciding whether to terminate a forward search, ISOPAR erroneously treats an indexed D as fixed, and initiates backward processing. It would probably be more logical to initiate QUITT although it would apparently make little actual difference in the result.

6.2. Changes to Improve Usability

Although the method of transferring the input symbolic code to the output cards through a Reproducer has been reasonably satisfactory, it is of course more troublesome than if the 533 would punch the entire information. This is impossible (1.3.1), but one of the changes which has been considered is to have columns 17-37 transferred to the output through the 533, so that the symbolic and assembled codes could be listed together for immediate check-out without waiting for reproducing. Normally, the remarks would be reproduced at a later time, unless the immediate check-out uncovered errors which warranted discarding the output code at this stage.

6.3. Changes to Speed Assembly And/Or Make More Drum Space Available

Most of the changes suggested in 6.1 and 6.2 require more drum space. There are several ways in which more space can be made available.

(a) ISOPAR uses 1900-1999 only during the read-in. Those cells are also used for tracing routines to hunt errors in ISOPAR, and for a punch-out routine to punch the code at the rate of 7 words per card. Those locations could not be used for instructions, but they could be used for part of the symbol table after the need for tracing is past.

(b) The sizes of the symbol table and of the storage region are flexible, and after any other changes they should be modified to fill the drum. If any change is made in the size or location of the storage region, note the following precautions, unless other revisions make them obsolete: The length of the storage region is a multiple of 5. The SYN cards for SMAXM and SMAXL must be corrected.

If a change is made in the size or location of the symbol table, note the following: The equivalence table is exactly half as long as the symbol table. The region table should immediately precede the symbol table, so that both can be cleared with one set-up, in Subroutine 1. Somewhere in the drum, there must be a zero word preceding the symbol table, and one following it; at present, these are in G0021 and ZMAX1. The SYN cards for ZMAXM and ZMAXL must be kept up to date. The constant ZTABL, which shows the length of the symbol table and appears at the end of the program, must be correct. The SYN card for HSYMB is used only to improve optimization, and only the dynamic level is significant; its dynamic level is at present 4 higher than that of ZMAXM.

(c) ISOPAR follows SOAP II in providing that the assembly routine uses only the basic 650 plus alphabetic feature; but that it will assemble programs involving other optional features. It would be better to prepare two different versions: Version A, which will assemble programs involving all optional features, but which uses index registers in the assembly; Version B, which omits all reference to index registers. Both of these would be considerably shorter than the present version; in fact, Version A would be several hundred words shorter. It seems very unlikely that a 650 without index registers would have to assemble a program that used them; if that should happen, the present version of ISOPAR could be used for the purpose.

(d) Another way of getting additional drum space (not over 60 locations) would be to provide that the instructions for handling core addresses be in locations 9000-9059, or as many of them as would be useful. This would not require separate codes for machines lacking

core memory, except for omitting the reading of the instructions that go into core; but it would mean that ISOPAR could process programs involving core only on a machine equipped with core memory.

T P	LOC	P S	OP	D ADDR	I ADDR	I ADDR	REMARKS, OR END COMMENTS *
1617	18-22	232425		26-30	31	32-36	37
K	0160	00	00	0000		0006	A constant, namely 6
Q	Q4		20	END		XXX	A comparison constant, to test for end of a loop
1	COMMENTS			CARD			A heading (comment) card
2	0050		30	F 0002		0007	Card of a relocatable subroutine, with fixed D
K-	0160		00	0000		0300	A negative constant, namely -300
			BLR	0306			To reserve 0306
			BLR	1296		1305	To reserve 1296-1305, inclusive
			RBD	0000			To reserve 0000-0049
			RBD	0600		1899	To reserve 0000 and 0600-1899
			60	X		AB	Blank L, left-justified symbolic D, right-justified symbolic I
	0I		24	016AC			Letter O, letter I, digit 0, digit 1, blank character
			21	0130			Absolute address
K	0350		10	1000		0051	Constant, namely 1010000051. 0350 is reserved, but not 1000 nor 0051
			20	X 0002			Store from Lower into word 2 of Region X
			REG	P 0177			Designate that P0001 = 0177, and reserve 0177
			REG	R 0101		0110	Designate that R0001 = 0101, and reserve 0101-0110
			20	P 0000	A		Store Lower into (P0000 + contents of index register A)

* On type I card, comments start in column 18.

FIGURE 1. Samples of ISOPAR input program cards

T P	LOC	P S	OP	D ADDR	I I ADDR	I A	REMARKS, OR END COMMENTS *
1617	18-22	232425		26-30	31	32-36 37	38-75
			20	F0002			F0002 denotes second word of region F
			REG	F0002			F0002 denotes that region F starts in location 0002
2			30	F0002		0007	F0002 denotes that 0002 is not to be modified in relocation
			REL	1200		0010	Type-2 addresses are relocated: Drum by 1200 and core by 0010
			REL			0020	Type-2 addresses are relocated: Drum unchanged and core by 0020
			REQ	SINEX		0000	Symbol SINEX = (0000 + amount of relocation shown by last REL card)
			RBR	0000		0051	Reserve 0000-0051 augmented by amount shown on last REL card
K	0013	ALF	BAX				Reserve 0013 and put 00 61 87 00 00 into it
K	PQ	ALF					Put positive zero into symbolic location PQ
K	0014	ALF					Reserve 0014 and put negative zero into it
		BLA	1699				Make 1699 available
		BLA	0169		0183		Make 0169-0183 available
		HED					Clear from symbol table all symbols which are not right-justified
		SYN	X		0100		Make right-justified symbolic X = 0100, and reserve 0100
		EQU	ALPHA		BETA		Make symbol ALPHA equal to value of symbol BETA, and reserve
		BOP					To assemble a second input program without rereading ISOPAR
		PAT					Punch availability table. Normally, used only at end of program
	8003	60		0100			A card with 800X in I can be used but is not useful

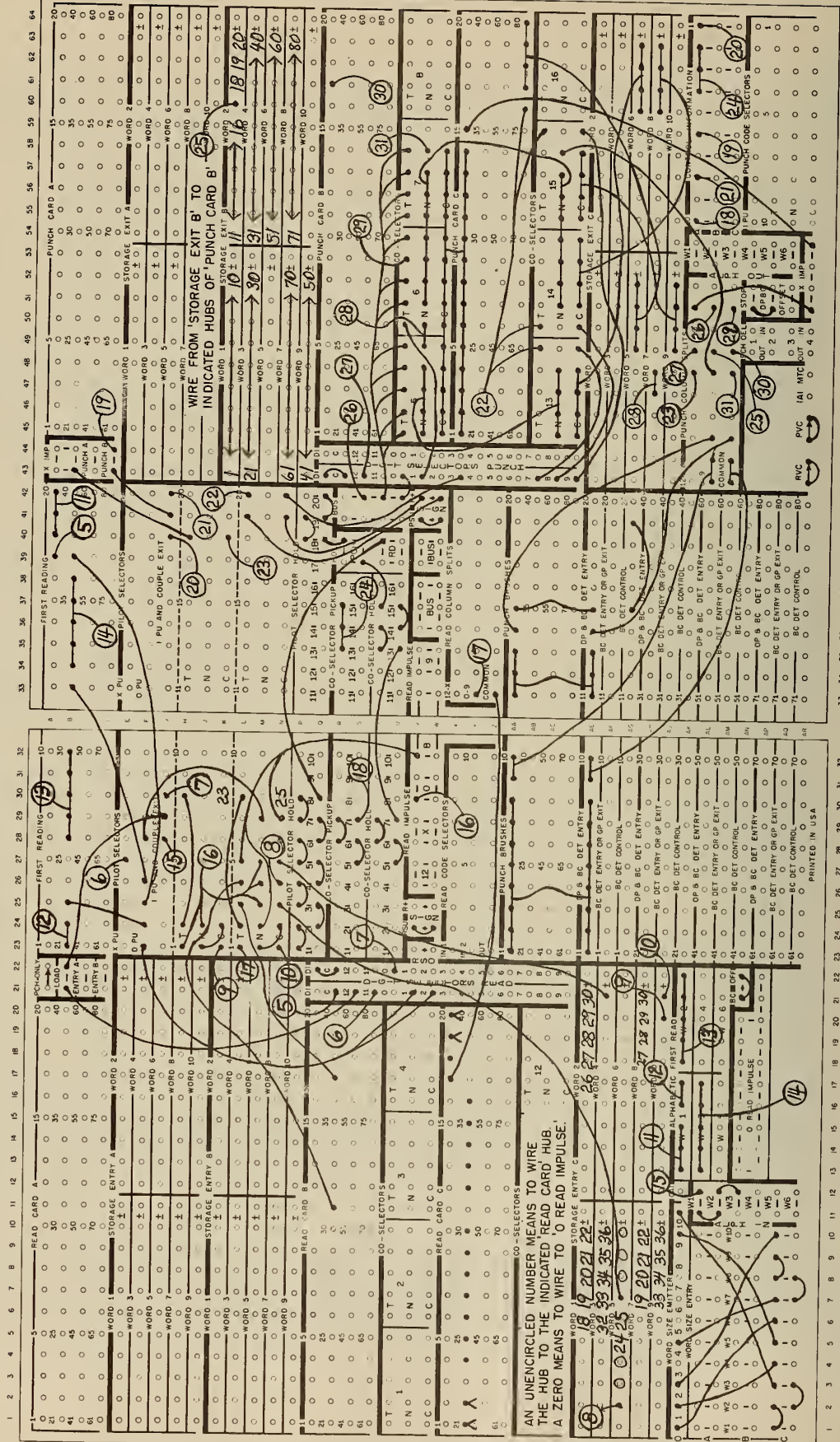
* On type I card, comments start in column 18.

FIGURE 2. Samples of ISOPAR input program cards

INTERNATIONAL BUSINESS MACHINES CORPORATION

BM 650 DATA PROCESSING SYSTEM

IBM 533-537 CARD READ PUNCH, CONTROL PANEL DIAGRAM



TYPE 533 ISOPAR CONTROL PANEL

S: WIRE TO INDICATED SECOND READING HUB
 N: WIRE TO INDICATED NORMAL PRINT ENTRY HUB
 T: WIRE TO INDICATED TRANSFER PRINT ENTRY HUB

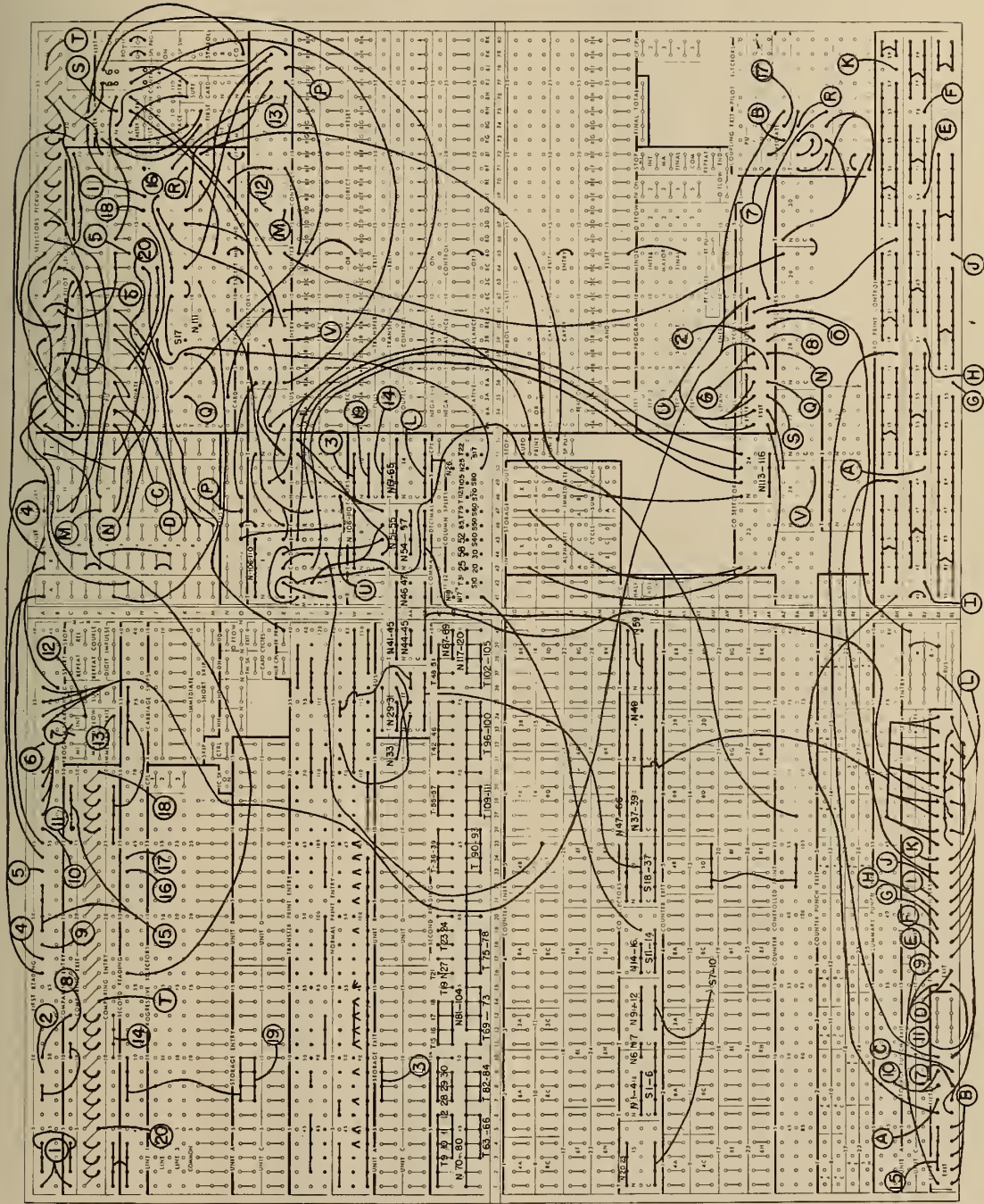


FIG. 4 TYPE 407 ISOPAR CONTROL PANEL

ROUTINE FOR THE IBM 650 DATA PROCESSING SYSTEM

4

RBD	1150		RESERVE PART OF REGION S	8
RBO	1350	1999		10
BLR	0002		TYPE 2 ROUTINE	12
BLR	0008		ORDINARY CARD	14
BLR	0010		MANUAL PAT	16
BLR	0016		ALF ROUTINE	18
BLR	0021		BLA ROUTINE	20
BLR	0025		SYN ROUTINE	22
BLR	0027		BOP ROUTINE	24
BLR	0029		BLR ROUTINE	26
BLR	0054		EOU ROUTINE	28
BLR	0073		PAT ROUTINE	30
BLR	0084		HCO ROUTINE	32
BLR	0093	0094	REL AND RBD ROUTINE	34
BLR	0097	0099	REG, REQ, RBR ROUTINES	36
REG	Q0118	0131	BRANCH ENTRIES IN EOU, SYN, AND REQ	38
REG	R0151	0160	READ REGION	40
REG	T0168	0172	ENTRIES IN PAT	42
REG	P0177	0186	PUNCH REGION	44
REG	F0218	0224	BRANCH ENTRIES IN SUBROUTINE 3	46
REG	M0268	0277	BRANCH ENTRIES IN SUBROUTINE 4	48
REG	J0368	0375	BRANCH ENTRIES IN SUBROUTINE 6	50
REG	L0418	0425	BRANCH ENTRIES IN SUBROUTINE 19	52
REG	O0469	0478	BRANCH ENTRIES IN SUBROUTINE 20	54
REG	X0518	0531	BRANCH ENTRIES IN BACKWARD PROCESSING	56
REG	I0572	0581	BRANCH ENTRIES IN SUBROUTINE 21	58
REG	U0608	0609	USEO IN HCO ROUTINE	60
BLR	0900	0903	CERTAIN TEMPORARIES	62
REG	C0918	0927	BRANCH ENTRIES IN SUBROUTINE 5	64
BLR	1030		SEARCH-START TAG	66
BLR	1050		FIRST-CARD TAG	68
REG	S1125	1149	STORAGE REGION, 1125-1199	70
BLA	1350	1354	RESERVED BY RBD, BUT NOT NEEDED	72
REG	A1355		AVAILABILITY TABLE, 1355-1554	74
REG	E1555		EQUIVALENCE TABLE, 1555-1659	76
REG	G1660		LIST OF REGIONS, 1660-1688	78
REG	Z1689		SYMBOL TABLE, 1689-1898	80
SYN	SETCC	0001	TYPE 1	82
SYN	HSYMB	0108	OPTIMIZE MAIN ITERATION OF SUBR 9	84
SYN	EXITX	P0001	TO CONSERVE DRUM SPACE	86
SYN	EXITY	P0002	TO CONSERVE DRUM SPACE	88
SYN	EXITZ	P0003	TO CONSERVE DRUM SPACE	90
SYN	DRUMT	G0010	TO CONSERVE DRUM SPACE	92
SYN	BLANB	G0020	TO CONSERVE DRUM SPACE	94
SYN	ZEROX	G0021	DOUBLE USE OF ZERO	96
SYN	SMAXM	1199	LAST CELL OF STORAGE AREA	98
SYN	SMAX1	1200	CELL BEYOND END OF STORAGE AREA	100
SYN	ZMAXM	1898		102
SYN	ZMAX1	1899	FOR CLEARING CELL BEYOND Z REGION	104
				106

OPTIMIZING ADDENDS AND TAGS

108

0800	00	0404	9998	K	0800	00	0404	9998	FOR OPERATION = LOCATION - 0800	110
0801	00	0404	9998	K	0801	00	0404	9998	FOR OPERATION = LOCATION - 0800	112
0802	33	2322	8999	K	0802	33	2322	8999	FOR OPERATION = LOCATION - 0800	114
0803	00	0505	8998	K	0803	00	0505	8998	FOR OPERATION = LOCATION - 0800	116
0804	00	0505	8998	K	0804	00	0505	8998	FOR OPERATION = LOCATION - 0800	118
0805	00	0505	8998	K	0805	00	0505	8998	FOR OPERATION = LOCATION - 0800	120
0806	00	0505	8998	K	0806	00	0505	8998	FOR OPERATION = LOCATION - 0800	122
0807	00	0505	8998	K	0807	00	0505	8998	FOR OPERATION = LOCATION - 0800	124
0808	33	1212	9999	K	0808	33	1212	9999	FOR OPERATION = LOCATION - 0800	126
0809	33	0202	9999	K	0809	33	0202	9999	FOR OPERATION = LOCATION - 0800	128
0810	33	0504	9999	K	0810	33	0504	9999	FOR OPERATION = LOCATION - 0800	130
0811	33	0504	9999	K	0811	33	0504	9999	FOR OPERATION = LOCATION - 0800	132
0812	99	9999	8999	K	0812	99	9999	8999	FOR OPERATION = LOCATION - 0800	134
0813	99	9999	8999	K	0813	99	9999	8999	FOR OPERATION = LOCATION - 0800	136
0814	33	1110	8999	K	0814	33	1110	8999	FOR OPERATION = LOCATION - 0800	138
0815	33	0504	9999	K	0815	33	0504	9999	FOR OPERATION = LOCATION - 0800	140
0816	33	0504	9999	K	0816	33	0504	9999	FOR OPERATION = LOCATION - 0800	142
0817	33	0504	9999	K	0817	33	0504	9999	FOR OPERATION = LOCATION - 0800	144
0818	33	0504	9999	K	0818	33	0504	9999	FOR OPERATION = LOCATION - 0800	146
0819	33	2120	8999	K	0819	33	2120	8999	FOR OPERATION = LOCATION - 0800	148
0820	54	0303	9999	K	0820	54	0303	9999	FOR OPERATION = LOCATION - 0800	150
0821	45	0303	9999	K	0821	45	0303	9999	FOR OPERATION = LOCATION - 0800	152
0822	34	0303	9999	K	0822	34	0303	9999	FOR OPERATION = LOCATION - 0800	154
0823	34	0303	9999	K	0823	34	0303	9999	FOR OPERATION = LOCATION - 0800	156

0824	33	0303	9999	K	0824	33	0303	9999	FOR OPERATION = LOCATION - 0800	158		
0825	44	0505	9999	K	0825	44	0505	9999	FOR OPERATION = LOCATION - 0800	160		
0826	00	0505	9998	K	0826	00	0505	9998	FOR OPERATION = LOCATION - 0800	162		
0827	00	0505	9998	K	0827	00	0505	9998	FOR OPERATION = LOCATION - 0800	164		
0828	33	1212	9999	K	0828	33	1212	9999	FOR OPERATION = LOCATION - 0800	166		
0829	33	0202	9999	K	0829	33	0202	9999	FOR OPERATION = LOCATION - 0800	168		
0830	00	0000	9988	K	0830	00	0000	9988	FOR OPERATION = LOCATION - 0800	170		
0831	00	0000	9888	K	0831	00	0000	9888	FOR OPERATION = LOCATION - 0800	172		
0832	33	2726	8999	K	0832	33	2726	8999	FOR OPERATION = LOCATION - 0800	174		
0833	33	2726	8999	K	0833	33	2726	8999	FOR OPERATION = LOCATION - 0800	176		
0834	33	2726	8999	K	0834	33	2726	8999	FOR OPERATION = LOCATION - 0800	178		
0835	00	0000	9988	K	0835	00	0000	9988	FOR OPERATION = LOCATION - 0800	180		
0836	00	0000	9988	K	0836	00	0000	9988	FOR OPERATION = LOCATION - 0800	182		
0837	33	2726	8999	K	0837	33	2726	8999	FOR OPERATION = LOCATION - 0800	184		
0838	33	2726	8999	K	0838	33	2726	8999	FOR OPERATION = LOCATION - 0800	186		
0839	33	2726	8999	K	0839	33	2726	8999	FOR OPERATION = LOCATION - 0800	188		
0840	33	0404	9998	K	0840	33	0404	9998	FOR OPERATION = LOCATION - 0800	190		
0841	33	0404	9998	K	0841	33	0404	9998	FOR OPERATION = LOCATION - 0800	192		
0842	33	0404	9998	K	0842	33	0404	9998	FOR OPERATION = LOCATION - 0800	194		
0843	33	0404	9998	K	0843	33	0404	9998	FOR OPERATION = LOCATION - 0800	196		
0844	34	0405	9998	K	0844	34	0405	9998	FOR OPERATION = LOCATION - 0800	198		
0845	43	0504	9998	K	0845	43	0504	9998	FOR OPERATION = LOCATION - 0800	200		
0846	33	0404	9998	K	0846	33	0404	9998	FOR OPERATION = LOCATION - 0800	202		
0847	33	0505	9998	K	0847	33	0505	9998	FOR OPERATION = LOCATION - 0800	204		
0848	33	0404	9998	K	0848	33	0404	9998	FOR OPERATION = LOCATION - 0800	206		
0849	33	0404	9998	K	0849	33	0404	9998	FOR OPERATION = LOCATION - 0800	208		
0850	00	0000	9988	K	0850	00	0000	9988	FOR OPERATION = LOCATION - 0800	210		
0851	00	0000	9988	K	0851	00	0000	9988	FOR OPERATION = LOCATION - 0800	212		
0852	00	0000	9988	K	0852	00	0000	9988	FOR OPERATION = LOCATION - 0800	214		
0853	00	0000	9988	K	0853	00	0000	9988	FOR OPERATION = LOCATION - 0800	216		
0854	44	0505	9999	K	0854	44	0505	9999	FOR OPERATION = LOCATION - 0800	218		
0855	00	0505	8998	K	0855	00	0505	8998	FOR OPERATION = LOCATION - 0800	220		
0856	00	0505	8998	K	0856	00	0505	8998	FOR OPERATION = LOCATION - 0800	222		
0857	00	0505	8998	K	0857	00	0505	8998	FOR OPERATION = LOCATION - 0800	224		
0858	00	0000	9988	K	0858	00	0000	9988	FOR OPERATION = LOCATION - 0800	226		
0859	00	0000	9988	K	0859	00	0000	9988	FOR OPERATION = LOCATION - 0800	228		
0860	33	0504	9999	K	0860	33	0504	9999	FOR OPERATION = LOCATION - 0800	230		
0861	33	0504	9999	K	0861	33	0504	9999	FOR OPERATION = LOCATION - 0800	232		
0862	99	9999	8999	K	0862	99	9999	8999	FOR OPERATION = LOCATION - 0800	234		
0863	99	9999	8999	K	0863	99	9999	8999	FOR OPERATION = LOCATION - 0800	236		
0864	33	1110	8999	K	0864	33	1110	8999	FOR OPERATION = LOCATION - 0800	238		
0865	33	0504	9999	K	0865	33	0504	9999	FOR OPERATION = LOCATION - 0800	240		
0866	33	0504	9999	K	0866	33	0504	9999	FOR OPERATION = LOCATION - 0800	242		
0867	33	0504	9999	K	0867	33	0504	9999	FOR OPERATION = LOCATION - 0800	244		
0868	33	0504	9999	K	0868	33	0504	9999	FOR OPERATION = LOCATION - 0800	246		
0869	33	0303	9999	K	0869	33	0303	9999	FOR OPERATION = LOCATION - 0800	248		
0870	00	0504	8999	K	0870	00	0504	8999	FOR OPERATION = LOCATION - 0800	250		
0871	00	0504	8999	K	0871	00	0504	8999	FOR OPERATION = LOCATION - 0800	252		
0872	00	0504	8999	K	0872	00	0504	8999	FOR OPERATION = LOCATION - 0800	254		
0873	00	0504	8999	K	0873	00	0504	8999	FOR OPERATION = LOCATION - 0800	256		
0874	00	0504	8999	K	0874	00	0504	8999	FOR OPERATION = LOCATION - 0800	258		
0875	00	0504	8999	K	0875	00	0504	8999	FOR OPERATION = LOCATION - 0800	260		
0876	00	0504	8999	K	0876	00	0504	8999	FOR OPERATION = LOCATION - 0800	262		
0877	00	0504	8999	K	0877	00	0504	8999	FOR OPERATION = LOCATION - 0800	264		
0878	00	0504	8999	K	0878	00	0504	8999	FOR OPERATION = LOCATION - 0800	266		
0879	00	0505	9999	K	0879	00	0505	9999	FOR OPERATION = LOCATION - 0800	268		
0880	01	0100	9988	K	0880	01	0100	9988	FOR OPERATION = LOCATION - 0800	270		
0881	00	0000	9988	K	0881	00	0000	9988	FOR OPERATION = LOCATION - 0800	272		
0882	01	0100	9988	K	0882	01	0100	9988	FOR OPERATION = LOCATION - 0800	274		
0883	00	0000	9988	K	0883	00	0000	9988	FOR OPERATION = LOCATION - 0800	276		
0884	33	2526	8999	K	0884	33	2526	8999	FOR OPERATION = LOCATION - 0800	278		
0885	00	0606	8998	K	0885	00	0606	8998	FOR OPERATION = LOCATION - 0800	280		
0886	00	0606	8998	K	0886	00	0606	8998	FOR OPERATION = LOCATION - 0800	282		
0887	00	0606	8998	K	0887	00	0606	8998	FOR OPERATION = LOCATION - 0800	284		
0888	01	0100	9988	K	0888	01	0100	9988	FOR OPERATION = LOCATION - 0800	286		
0889	00	0000	9988	K	0889	00	0000	9988	FOR OPERATION = LOCATION - 0800	288		
0890	44	0505	9998	K	0890	44	0505	9998	FOR OPERATION = LOCATION - 0800	290		
0891	33	0505	9998	K	0891	33	0505	9998	FOR OPERATION = LOCATION - 0800	292		
0892	33	0505	9998	K	0892	33	0505	9998	FOR OPERATION = LOCATION - 0800	294		
0893	33	0505	9998	K	0893	33	0505	9998	FOR OPERATION = LOCATION - 0800	296		
0894	33	0505	9998	K	0894	33	0505	9998	FOR OPERATION = LOCATION - 0800	298		
0895	33	0505	9998	K	0895	33	0505	9998	FOR OPERATION = LOCATION - 0800	300		
0896	33	0505	9998	K	0896	33	0505	9998	FOR OPERATION = LOCATION - 0800	302		
0897	33	0505	9998	K	0897	33	0505	9998	FOR OPERATION = LOCATION - 0800	304		
0898	33	0505	9998	K	0898	33	0505	9998	FOR OPERATION = LOCATION - 0800	306		
0899	44	0505	9998	K	0899	44	0505	9998	FOR OPERATION = LOCATION - 0800	308		
1250	23	2225	2400	2225	K	1250	23	2225	2400	2225	TABLE TO OPTIMIZE SHIFT OPERATIONS	310
1251	07	0607	0600	K	1251	07	0607	0600	0800	TABLE TO OPTIMIZE SHIFT OPERATIONS	312	
1252	07	0609	0800	K	1252	07	0609	0800	1000	TABLE TO OPTIMIZE SHIFT OPERATIONS	314	
1253	09	0811	1000	K	1253	09	0811	1000	1200	TABLE TO OPTIMIZE SHIFT OPERATIONS	316	
1254	11	1013	1200	K	1254	11	1013	1200	1400	TABLE TO OPTIMIZE SHIFT OPERATIONS	318	
1255	13	1215	1400	K	1255	13	1215	1400	1600	TABLE TO OPTIMIZE SHIFT OPERATIONS	320	
1256	15	1417	1600	K	1256	15	1417	1600	1800	TABLE TO OPTIMIZE SHIFT OPERATIONS	322	
1257	17	1619	1800	K	1257	17	1619	1800	2000	TABLE TO OPTIMIZE SHIFT OPERATIONS	324	
1258	19	1821	2000	K	1258	19	1821	2000	2200	TABLE TO OPTIMIZE SHIFT OPERATIONS	326	
1259	21	2023	2200	2023	K	1259	21	2023	2200	2023	TABLE TO OPTIMIZE SHIFT OPERATIONS	328
1300	19	9906	0690	K	1300	19	9906	0690	0880	TABLE TO OPTIMIZE INDEXING OPERATIONS	330	
1301	80	0008	0880	K	1301	80	0008	0880	0680	TABLE TO OPTIMIZE INDEXING OPERATIONS	332	
1302	80	0106	0680	K	1302	80	0106	0680	0209	TABLE TO OPTIMIZE INDEXING OPERATIONS	334	
1303	80	0209	0880	K	1303	80	0209	0880	0308	TABLE TO OPTIMIZE INDEXING OPERATIONS	336	
1304	80	0308	0980	K	1304	80	0308	0980		TABLE TO OPTIMIZE INDEXING OPERATIONS	338	

1305 99 9908 0880	K	1305 99 9908 0880	TABLE TO OPTIMIZE INOEXING OPERATIONS	340
-------------------	---	-------------------	---------------------------------------	-----

OTHER CONSTANTS WITH ABSOLUTE LOCATIONS	342
---	-----

1200 19 9900 0000	K	1200 19 9900 0000	IF DRUM ADDRESS, USE REGULAR EXIT	344
1201 79 9900 0004	K	1201 79 9900 0004	IF NOT AN ADDRESS, ADD 4	346
1202 80 0300 0002	K	1202 80 0300 0002	IF 800X ADDRESS, ADD 2	348
1203 80 0400 0004	K	1203 80 0400 0004	IF NOT AN ADDRESS, ADD 4	350
1204 80 0700 0002	K	1204 80 0700 0002	IF 800X ADDRESS, ADD 2	352
1205 89 9900 0004	K	1205 89 9900 0004	IF NOT AN ADDRESS, ADD 4	354
1206 90 5900 0003	K	1206 90 5900 0003	IF CORE ADDRESS, ADD 3	356
1207 99 9999 0004	K	1207 99 9999 0004	IF NOT AN ADDRESS, ADD 4	358

MAIN ROUTINE	360
--------------	-----

0100 69 0003 0006	N	0100 69 READC	SUBR1	INITIALIZE AT START OF ASSEMBLY	362
0003 70 0161 1041		READC 70 R0011		READ ONE CARD	364
1041 69 1044 1047		69 81STX		RESTORE FIRST CARD TAG)	366
1047 24 1050 0160	1050	24 1050	R0010	1050 RESTORE FIRST CARD TAG) MULTIBRANCH	368

ORDINARY CARO	370
---------------	-----

0008 69 1044 0047	N	0008 69 81STX		DELETE QUITT TAG)	372
0047 24 1030 0033	1030	24 1030		DELETE QUITT TAG)	374
0033 69 0036 0039		69	SUB14	OP, C.I., OPTIM, ALOPT, ETC.	376
0036 69 0089 0042	J	69	SUB19	PROCESS L	378
0089 69 0092 0045	J	69	SUB20	PROCESS 0	380
0092 69 0095 0048	J	69 OPTIM		IS IT AN MOF OPERATION)	382
0048 94 0051 0053		94 MDFLP		D, YES	384
0053 60 0902 0007		60 0902		IS 0 INOEXEO	386
0007 44 0011 0012		44 MDFL1	ABCDE	IF SO, START FORWARD SEARCH	388
0012 69 0015 0018	C ABCDE	69	SUB21	PROCESS I	390
0015 60 0003 0057	J	60 READC	SB10A	PREPARE TO PUNCH CARD	392

FORWARD SEARCH	394
----------------	-----

HED	396
-----	-----

0051 69 0004 0107	N MOFLP	69 P	A	STARTED SEARCH WITH MOF OPERATION	398
0011 69 0014 0107	N MOFL1	69 I	A	STARTED SEARCH WITH INOEXEO	400
1001 69 1004 0107	N MDFL1	69 L	A	STARTED WITH NEW-SYMBOL L	402
0107 24 1030 0083	1030	24 1030		STORE TAG TELLING HOW SEARCH STARTED	404
0083 65 0086 0041	C A	65 SEND	LOOP	STORE SET OF DATA)	406
0041 10 0044 8003	C LOOP	10	8003	STORE SET OF DATA)	408
0044 69 0151 8002	J	69 R0001	8002	STORE SET OF DATA)	410
0086 24 1125 0028	1125	24 S0001	X	S0001 STORE SET OF DATA)	412
0028 11 0031 0035	J SEND	11 0		STORE SET OF DATA)	414
0035 15 0038 0043	X	15 16THX		STORE SET OF DATA)	416
0043 44 0147 0148		44	OUT	STORE SET OF DATA)	418
0147 10 0050 8003		10 QS	8003	STORE SET OF DATA)	420
0148 20 0103 0056	0103	20 FINAL		FINAL STORE VARIABLE ORDER	422
0056 69 1050 0203	N OUT	69 1050		IS THIS FIRST SET OF SEARCH)	424
0203 90 0207 0058		90	NO	IS THIS FIRST SET OF SEARCH)	426
0207 69 1030 0133		69 1030		IF SO, WHERE DID SEARCH START	428
0133 96 0136 0088		96 B	C	WITH I OR L, RESPECTIVELY	430
0058 65 0160 0065	N NO	65 R0010		IS CARO OF TYPE 08)	432
0065 16 0068 0023		16 810TH		IS CARD OF TYPE 08)	434
0023 45 0026 0077		45 QUITT		D, IT IS NOT, AND WE QUIT	436
0077 60 0151 0005		60 R0001		ALPHABETIC L	438
0005 69 0208 0061		69	SUBR3	IS L FIXED ADDRESS)	440
0208 35 0005 0071	J	35 0005		IS L FIXED ADDRESS)	442
0071 44 0026 0076		44 QUITT		IF SO, QUIT SEARCH)	444
0076 45 0088 0026		45 C	QUITT	IF SO, QUIT SEARCH)	446
0088 69 0091 0039	C C	69	SUB14	GET Op, C.I., TAGS, OPTIM, ALOPT.	448

0091 95 0144 0046	J	95 K		IS 0 ACTUALLY AN ADDRESS	450
0046 60 0152 0257		60 R0002		IF SO, IS IT FIXED)	452
0237 69 0060 0061		69	SUBR3	IF SO, IS IT FIXED)	454
0060 44 0026 0064	J	44 QUITT		IF N G, QUIT SEARCH	456
0064 45 0144 0019		45 K	BACKW	IF FIXED, START BACKWARD PROCESSING	458
0144 69 0197 0150	C K	69 ALOPT		IS IT AN MOF OPERATION	460
0150 94 0026 0055		94 QUITT		IF SO, QUIT SEARCH	462
0055 60 0902 0307		60 0902		IS 0 INOEXED	464
0307 45 0026 0136		45 QUITT	B	IF SO, QUIT SEARCH	466
0136 60 0153 0357	C B	60 R0003		IS I A FIXED ADDRESS)	468
0357 69 0110 0061		69	SUBR3	IS I A FIXED ADDRESS)	470
0110 35 0005 0173	J	35 0005		IS I A FIXED ADDRESS)	472
0173 44 0026 0078		44 QUITT		IF I IS N G, QUIT SEARCH	474
0078 45 0032 0019		45 BC0EF	BACKW	IF I IS FIXED, START BACKWARD	476
0032 65 0103 0407	N BCDEF	65 FINAL		IS STORAGE AREA FULL)	478
0407 16 0210 0115		16 MAX		IS STORAGE AREA FULL)	480
0115 45 0318 0026		45	QUITT	IF SO, QUIT SEARCH	482
0318 65 0903 0457		65 0903		IS I INOEXED	484
0457 45 0026 0111		45 QUITT		IF SO, QUIT SEARCH	486
0111 69 0114 0017		69 91STX		CHANGE FIRST-CARD INDICATOR TO 2ND)	488
0017 24 1050 0253	1050	24 1050		CHANGE FIRST-CARD INDICATOR TO 2ND)	490
0253 70 0151 1007		70 R0001		READ ANOTHER CARD	492
1007 65 0160 0165		65 R0010		IS IT TYPE 08)	494
0165 16 0068 0323		16 810TH		IS IT TYPE 08)	496
0323 45 0176 0227		45	LP	O, IT IS NOT. I, IT IS,	498
0176 15 0079 0233		15 710		IS TYPE 01	500
0233 45 0236 0037		45 YYY		D, NO. I, YES.	502
0037 24 0151 0104	0151	24 R0001		R0001 IF SO, PUT IN ODDMY MOVEABLE WORDS)	504
0104 24 0152 0105	0152	24 R0002		R0002 IF SO, PUT IN ODDMY MOVEABLE WORDS)	506
0105 24 0153 0236	0153	24 R0003	YYY	R0003 IF SO, PUT IN ODDMY MOVEABLE WORDS)	508
0236 24 0155 0227	0155	24 R0005	LP	R0005 EITHER CASE, SOMETHING INTO R0005	510
0227 65 0103 0041	C YYY C LP	65 FINAL	LOOP	ALL CASES, BACK TO START OF LOOP	512
0031 69 0155 8002	O O	69 R0005	8002	COMPARISON CONSTANT FOR STORING	514
0050 69 0156 8002	O OS	69 R0006	8002	TO RESTORE COMPARISON CONST + ADVANCE	516
0210 24 1200 0028	1200 Q MAX	24 SMAX1	X	SMAX1 CONSTANT FOR WHETHER STORAGE IS FULL	518
0004 90 0088 0000	K P	90 0088	0000	TAG FOR SEARCH STARTED BY MOF	520
0014 90 0089 0000	K I	90 0089	0000	TAG FOR SEARCH STARTED BY INOEXED 0	522
1004 90 0090 0000	K L	90 0090	0000	TAG, SEARCH STARTED BY NEW-SYMBOL L	524
0079 00 0000 0007	K 710	00 0000	0007		526

QUIT WITHOUT FIXED ADDRESS

HED					528
					530
0026 69 1050 0303	N QUITT	69 1050		WAS THERE ONLY ONE SEARCHED CARD	532
0303 90 0507 0258		90	SKP	D, ONLY ONE.	534
0507 69 1030 0283		69 1030		WHERE DID SEARCH START	536
0283 96 0012 0138		96 ABCDE		IF WITH I, PROCESS LIKE 08	538
0138 69 0141 0042		69	SUB19	PROCESS L	540
0141 69 0012 0045	J	69 ABCDE	SUB20	PROCESS 0, AND TO 08 ROUTINE	542
0238 65 0103 0557	N SKP	65 FINAL		NOT 1ST CARD. MAKE COMP. CONST.)	544
0557 10 0260 0215		10 BRNG		NOT 1ST CARD. MAKE COMP. CONST.)	546
0215 22 0069 0022	0069	22 OC		NOT 1ST CARD. MAKE COMP. CONST.)	548
0022 65 8003 0229		65 8003	LOOP	BRING BACK A SET)	550
0229 10 0082 8002	C LOOP	10 SENO	8002	BRING BACK A SET)	552
0260 69 1125 8003	J BRNG	69 S0001	8003	BRING BACK A SET)	554
0082 24 0151 0204	0151 J SENO	24 R0001	X	R0001 BRING BACK A SET)	556
0204 11 0607 0161	X	11 O		BRING BACK A SET)	558
0161 15 0038 0143		15 16THX		BRING BACK A SET)	560
0143 44 0247 0198		44	OUT	BRING BACK A SET)	562
0247 10 0200 8002		10 QS	8002	BRING BACK A SET)	564
0198 20 0353 0106	0353 N OUT	20 VAR		STORE VARIABLE ORDER	566
0106 16 0069 0623		16 QC		IS THIS LAST SET	568
0623 45 0226 0160		45	R0010	IF SO, MULTIBRANCH + EXIT FROM QUITT	570
0226 69 0279 0039		69	SUB14	GET OP, C.I., TAGS, OPTIM, ALOPT.	572
0279 69 1050 0403	J	69 1050		IS THIS FIRST QUITT SET	574
0403 90 0657 0308		90 NO		O, IT IS NOT	576
0308 69 1030 0333		69 1030		IF SO, WHERE DID WE START	578
0333 96 0286 0657		96 JUMP	NO	D, STARTED WITH I. I, WITH L.	580
0657 69 0310 0042	C NO	69	SUB19	STARTED WITH L, OR NOT FIRST. PROC. L	582

D310 69 0286 0045	J	69 JUMP	SUB20	PROCESS D	584
D286 69 D139 0018	J JUMP	69	SUB21	PROCESS I	586
0139 69 1044 0297	J	69 81STX		CHANGE TAG TO SECOND-CARD TAG I	588
0297 24 1050 0453	1050	24 105D		CHANGE TAG TO SECOND-CARD TAG I	590
D453 60 0206 0057		60	SB10A	PUNCH CARD	592
0206 65 0353 0229	J	65 VAR	LOOP	BRING VARIABLE ORDER, AND REPEAT	594
D6D7 24 D155 02D4	D155	24 RDD05	X	RDD05 COMPARISON CONSTANT FOR BRINGING SET	596
0200 24 0156 02D4	D156	24 RD006	X	RD006 TO RESTORE AND ADVANCE	598

BACKWARDS ROUTINE

HEO					60D
					602
0019 69 1050 0503	N BACKW	69 105D		WAS THERE JUST ONE SEARCH CARD	604
0503 90 07D7 0358		9D	A	IF NOT, JUMP AHEAD	606
0707 69 103D 0383		69 103D		IF SO, WHERE DID SEARCH START	608
0383 96 D336 0188		96	B	D, WITH I. I, WITH L.	610
0336 95 0032 0012		95 BCDEF	ABCDE	WITH I. EXIT ACCORDING TO WHY SEARCH	612
0188 69 0114 0067	N B	69 91STX		START ON L. FIRST-CARD TAG I	614
0D67 24 1D5D 0358	1D5D	24 1D5D	A	START ON L. FIRST-CARD TAG I	616
0358 69 0211 0164	C A	69 SEN		PRESET TO STORE AFTER PROCESSING I	618
0164 24 0117 0D20	D117	24 VAR2X		PRESET TO STORE AFTER PROCESSING I	620
0D2D 69 0182 D085		69 PDD06		SAVE L FROM CARD THAT STARTED SRCH I	622
0D85 24 D238 0191	D238	24 SAVEL		SAVE L FROM CARD THAT STARTED SRCH I	624
D191 69 0184 D087		69 P0008		SAVE D FROM CARD THAT STARTED SRCH I	626
0D87 24 004D 0193	DD40	24 SAVED		SAVE D FROM CARD THAT STARTED SRCH I	628
D193 69 0D96 0D49		69 DN1		PRESET EXIT FROM BACKWARDS ROUTINE I	63D
0D49 24 0D52 0D05	DD52	24 DONE		PRESET EXIT FROM BACKWARDS ROUTINE I	632
D205 65 0103 0757		65 FINAL		BRING BACK A SET I	634
0757 69 D36D 0D13		69 BRNG		BRING BACK A SET I	636
0013 22 0167 D070	D167	22 XXXX1		BRING BACK A SET I	638
0070 65 8D01 0327		65 8DD1		BRING BACK A SET I	64D
0327 16 DD38 0243		16 16THX	LOOP	BRING BACK A SET I	642
0360 69 9999 8DD3	J BRNG	69 9999	8DD3	BRING BACK A SET I	644
D243 10 D146 8DD2	N LOOP	1D	8DD2	BRING BACK A SET I	646
D146 24 D155 D408	D155	24 RDD05	X	RDD05 BRING BACK A SET I	648
D408 11 D261 0265	X	11 Q		BRING BACK A SET I	65D
D265 16 0038 D293		16 16THX		BRING BACK A SET I	652
D293 44 0347 D248		44	OUT	BRING BACK A SET I	654
D347 10 0250 8DD2		10 QS	8DD2	BRING BACK A SET I	656
0248 2D 0553 0256	D553	2D VAR		STORE VARIABLE ORDER	658
D256 69 0D09 0D39	N OUT	69	SUB14	GET Op. C.I., TAGS, OPTIM, ALOPT.	66D
0009 65 0553 09D7	J	65 VAR		IS THIS LAST SET I	662
D907 16 041D 0315		16 QF		IS THIS LAST SET I	664
0315 45 0468 0319		45 NO		D, NO. I, YES.	666
D319 69 1D3D D433		69 1D3D		IF SO, WHERE DID SEARCH START	668
D433 96 0386 0288		96	L	D, WITH I. I, WITH L.	67D
D386 69 D189 D142		69	SUBR6	PROCESS I BACKWARD	672
0189 69 0238 0241	J	69 SAVEL		BRING BACK L AND D I	674
0241 24 0182 0135	D182	24 PDD06	P0006	BRING BACK L AND D I	676
0135 69 DD40 0343		69 SAVED		BRING BACK L AND D I	678
D343 24 0184 0137	D184	24 PDD08	ENDED	PDD08 BRING BACK L AND D I	68D
D288 69 0137 0D9D	N L	69 ENDED		WITH L. ALTER EXIT I	682
D09D 24 0052 0468	D052	24 DONE	NO	WITH L. ALTER EXIT I	684
D468 69 D197 03D0	C NO	69 ALOPT		IS D AN ACTUAL ADDRESS	686
D300 95 0603 0255		95 SHX		D, NO. I, YES.	688
D255 6D 0153 0957		6D RDD03		BRING ALPHABETIC I	69D
D957 69 046D D063		69	SUBR9	ANALYZE I, FOR TYPE OF ADDRESS	692
D46D 65 D197 052D	J	65 ALOPT	XDD03	MULTIPLE EXIT, ACCORDING TO KIND	694
D522 15 D075 0329	R X0005	15 13RD	X1	8DDX. CALL IT UNFIXED	696
0518 15 D075 D329	R XDD01	15 13RD	X1	ADDRESS N G. CALL UNFIXED	698
D519 69 1D5D D653	R X0002	69 1D5D		BLANK ADDRESS. IS THIS FIRST CARD	7D0
0653 9D 0329 0521		9D X1	XDD04	FIRST, UNFIXED. OTHER, FIXED.	7D2
D52D DD DD0D 0329	R XDD03	DD DD0D	X1	DRUM, OLD SYMB, REG. FIXED	704
D521 15 0D75 0329	R XD004	15 13RD	X1	NEW SYMBOL. UNFIXED	706
0523 0D DD0D 0519	R XD006	0D DD0D	X0002	CORE. TREAT SAME AS BLANK	7D8

0524 00 0000 0329		R X0007	00 0000	X1		OTHER ADDRESS. FIXED	710
0329 20 0197 0350	0197	C X1	20 ALOPT		ALOPT	STORE MODIFIED ALOPT	712
0350 60 0152 1057			60 R0002			ALPHABETIC D, TO SEE WHETHER FIXED	714
1057 69 0510 0063			69	SUBR9		ANALYZE D	716
0510 65 0197 0527		J	65 ALOPT	X0010		MULTIPLE EXIT	718
0529 15 0132 0187		R X0012	15 12NDX	X2		800X ADDRESS. UNFIXED	720
0525 15 0132 0187		R X0008	15 12NDX	X2		D IS N G. CALL UNFIXED	722
0526 69 1050 0703		R X0009	69 1050			BLANK. IS THIS FIRST CARD	724
0703 90 0187 0528			90 X2	X0011		NO, FIXED. YES, UNFIXED.	726
0527 00 0000 0081		R X0010	00 0000	F		OLD SYMBOL, DRUM, REGION. FIXED	728
0528 15 0132 0187		R X0011	15 12NDX	X2		NEW SYMBOL. UNFIXED	730
0530 91 0526 0525		R X0013	91 X0009	X0008		CORE. BRANCH, LIKE BLANK. ARITH, UNF	732
0531 01 0666 0531		R X0014	01 0666	X0014		OTHER. ERROR. SHOULD NOT BE HERE.	734
0187 20 0197 0081	0197	N X2	20 ALOPT	F	ALOPT	STORE MODIFIED ALOPT	736
0081 91 0034 0436		C F	91 XX8			MULTIPLE BRANCH ACCORDING TO TAGS)	738
0436 99 0140 0291			99	MISC		MULTIPLE BRANCH ACCORDING TO TAGS)	740
0140 98 0291 0145			98 MISC	899		MULTIPLE BRANCH ACCORDING TO TAGS)	742
0034 99 0338 0239		N XX8	99	9X8		MULTIPLE BRANCH ACCORDING TO TAGS)	744
0338 98 0341 0393			98 888	898		MULTIPLE BRANCH ACCORDING TO TAGS)	746
0261 20 0151 0408	0151	Q Q	24 R0001	X		R0001 COMPARISON CONSTANT FOR END OF SET	748
0250 20 0150 0408	0150	Q QS	24 R0000	X		R0000 RESTORE COMPARISON CONSTANT AND MODIFY	750
0410 69 1124 8003		Q QF	69 S0000	8003			752
0075 00 1000 0000		K 13RD	00 1000	0000			754

FIXED I, OR BOTH UNFIXED, ARITHMETIC

0291 69 0194 0142		N MISC	69	SUBR6		PROCESS I BACKWARDS	756
0194 69 0397 0400		J	69 LNM	SUBR5		PROCESS D BACKWARDS	760
0397 69 0052 0305		J LNM	69 DONE	SUBR4		PROCESS L BACKWARDS	762

SHIFT OR INDEXING OPERATION

0603 69 0306 0045		N SHX	69	SUB20		PROCESS D FORWARDS	764
0306 69 0397 0142		J	69 LNM	SUBR6		PROCESS I BACKWARDS	768

UNFIXED I, FIXED D, ARITHMETIC

0145 69 0298 0400		N 899	69	SUBR5		PROCESS D BACKWARDS	772
0298 69 0101 0018		J	69	SUB21		PROCESS I FORWARDS	774
0101 65 0397 0201		J	65 LNM	SUB22		SAVE ORCEQ	776

UNFIXED I, FIXED D, BRANCH OPERATION

0393 69 0196 0400		N 898	69	SUBR5		PROCESS D BACKWARDS	780
0196 69 0149 0305		J	69	SUBR4		PROCESS L BACKWARDS	782
0149 69 0102 0018		J	69 LN	SUB21		PROCESS I FORWARDS	784
0102 65 0052 0201		J LN	65 DONE	SUB22		SAVE ORCEQ	786

UNFIXED D, BRANCH OPERATION

0239 69 0192 0142		N 9X8	69	SUBR6		PROCESS I BACKWARDS	790
-------------------	--	-------	----	-------	--	---------------------	-----

0192 69 0195 0305	J	69	SUBR4	PROCESS L BACKWARDS	792
0195 69 0102 0045	J	69 LN	SUB20	PROCESS D FORWARDS	794

FIXED D AND I, BRANCH OPERATION

0341 69 1679 0232	N 888	69 BLANB			798
0232 24 0235 0388	0235	24 T1	T1	SAVE IN CASE I IS BLANK	800
0388 69 0391 0244		69 ORCEB		SAVE IN CASE I IS BLANK	802
0244 24 0447 0450	0447	24 T2	T2	SAVE IN CASE I IS CORE)	804
0450 65 0197 0251		65 ALOPT		ALOFT TO LOWER	806
0251 16 0254 0059		16 11STX		CHANGE ITS FIRST DIGIT TO 8	808
0059 20 0197 0500	0197	20 ALOPT	ALOFT	STORE MODIFIED ALOFT	810
0500 69 0753 0400		69	SUBR5	PROCESS D BACKWARDS	812
0753 69 0356 0305	J	69	SUBR4	PROCESS L BACKWARDS FROM 0	814
0356 65 0109 0113	J	65	SSB	BRING BACK BLANB AND ORCEB	816
0109 65 0062 0217	J	65 OPREG		SAVE DYNAMIC LEVEL OF L FROM 0)	818
0217 14 0320 1036		14 501XX		SAVE DYNAMIC LEVEL OF L FROM D)	820
1036 21 1091 1094	1091	21 OLO	OLO	SAVE DYNAMIC LEVEL OF L FROM 0)	822
1094 65 0197 0301		65 ALOPT		MODIFY TO USE ROUTINE FOR UNFIXED 0)	824
0301 15 0132 0237		15 12NOX		MODIFY TO USE ROUTINE FOR UNFIXED D)	826
0237 20 0197 0550	0197	20 ALOPT	ALOFT	MODIFY TO USE ROUTINE FOR UNFIXED D)	828
0550 69 0953 0142		69	SUBR6	PROCESS I BACKWARDS	830
0953 69 0406 0305	J	69	SUBR4	PROCESS L BACKWARDS FROM I	832
0406 65 0062 0267	J	65 OPREG		WHICH L IS LESS, MEASURED ON CIRCLE)	834
0267 14 0320 1080		14 501XX		WHICH L IS LESS, MEASURED ON CIRCLE)	836
1080 65 8003 1038		65 8003		WHICH L IS LESS, MEASURED ON CIRCLE)	838
1038 16 1091 0245		16 DLO		WHICH L IS LESS, MEASURED ON CIRCLE)	840
0245 46 0348 0199		46 AEB		WHICH L IS LESS, MEASURED ON CIRCLE)	842
0199 16 0202 1107		16 251XX	BOT	WHICH L IS LESS, MEASURED ON CIRCLE)	844
0348 15 0202 1107	N AEB	15 251XX	BOT	WHICH L IS LESS, MEASURED ON CIRCLE)	846
1107 46 0560 0311	C BOT	46	2NO	D, WE WILL USE ONE MEASURED FROM D	848
0560 65 0163 0113		65	SSB	BRING BACK BLANB AND ALOFT ONCE MORE	850
0163 69 0066 0400	J	69	SUBR5	PROCESS BACKWARDS 0 ONCE MORE	852
0066 65 0197 0351	J	65 ALOPT		MODIFY ALOFT AGAIN)	854
0351 16 0132 0287		16 12NOX	XY	MODIFY ALOFT AGAIN)	856
0287 15 0254 0209	C XY	15 11STX		MODIFY ALOFT AGAIN)	858
0209 20 0197 0397	0197	20 ALOPT	LNH	ALOFT STORE NEW ALOFT, AND JUMP BACK	860
0311 65 0214 0113	N 2ND	65	SSB	USE I-COMP. BRING BLANB AND ORCEB	862
0214 69 0317 0142	J	69	SUBR6	PROCESS I BACKWARDS AGAIN	864
0317 65 0197 0287	J	65 ALOPT	XY	BRING ALOFT, AND JUMP BACK	866
0113 69 0235 0438	N SSB	69 T1		SPEC. SUBR. BRING BACK BLANB + ORCEB	868
0438 24 1679 0282	1679	24 BLANB	BLANB	SPEC. SUBR. BRING BACK BLANB + ORCEB	870
0282 69 0447 0600		69 T2		SPEC. SUBR. BRING BACK BLANB + ORCEB	872
0600 24 0391 8002	0391	24 ORCEB	8002	SPEC. SUBR. BRING BACK BLANB + ORCEB	874
0096 65 0117 0321	J ONI	65 VAR2X		ALL CASES. STORE PROCESSED CARD)	876
0321 10 0024 8003		10	8003	STORE PROCESSED CARD)	878
0024 69 0186 8002	J	69 P0010	8002	STORE PROCESSED CARD)	880
0211 24 1199 0252	1199	24 SMAXH	SMAXH	STORE PROCESSED CARD)	882
0252 11 0355 0259		11 02		STORE PROCESSED CARD)	884
0259 16 0038 0443		16 16THX		STORE PROCESSED CARD)	886
0443 44 0497 0398		44	OUT2	STORE PROCESSED CARD)	888
0497 10 0650 8003		10 QS2	8003	STORE PROCESSED CARD)	890
0398 20 0117 0570	0117	20 VAR2X	VAR2X	STORE PROCESSED CARD)	892
0570 65 0553 0243		65 VAR	LOOP	BRINGING ORDER. BACK FOR ANOTHER CARD	894
0355 69 0182 8002	Q 02	69 P0006	8002	COMPARISON CONSTANT FOR END OF SET	896
0650 69 0181 8002	Q QS2	69 P0005	8002	RESTORE CONSTANT AND MODIFY	898

BACKWARDS PUNCH

0137 65 0117 0571	J ENOED	65 VAR2X		MAKE NEW BRINGING ORDER)	904
0571 69 0074 0377		69 BRNG		MAKE NEW BRINGING ORDER)	906
0377 22 0167 0620	0167	22 XXXX1	LOOP	XXXX1 MAKE NEW BRINGING ORDER)	908

0620 60 0673 0057	C LOOP	60	SB10A	TO SUB 10A TO PUNCH ONE CARD	910
0673 65 0167 0621	J	65 XXXX1		BRING BACK A SET OF RESULTS)	912
0621 16 0174 0379		16 OL		BRING BACK A SET OF RESULTS)	914
0379 45 0332 0483		45	OVR	BRING BACK A SET OF RESULTS)	916
0332 15 8001 0289		15 8001		BRING BACK A SET OF RESULTS)	918
0289 10 0242 0547		10 SEND	A	BRING BACK A SET OF RESULTS)	920
0547 15 0038 8002	C A	15 16THX	8002	BRING BACK A SET OF RESULTS)	922
0074 69 9999 8003	J BRNG	69 9999	8003	BRING BACK A SET OF RESULTS)	924
0242 24 0182 0285	0182 J SEND	24 P0006	X	P0006 BRING BACK A SET OF RESULTS)	926
0285 11 0488 0493	X	11 O		BRING BACK A SET OF RESULTS)	928
0493 44 0597 0448		44	OUT	BRING BACK A SET OF RESULTS)	930
0597 10 0700 0547		10 OS	A	BRING BACK A SET OF RESULTS)	932
0448 20 0167 0620	0167 N OUT	20 XXXX1	LOOP	XXXX1 STORE MODIFIED BRINGING ORDER + REPEAT	934
0483 69 0486 0339	N OVR	69 SAVOR		FINISHED. SAVE SAVOR AS ORCEO)	936
0339 24 0292 0003	0292	24 ORCEO	READC	ORCEO FINISHED. SAVE SAVOR AS ORCEO)	938
0488 24 0186 0285	0186 O O	24 P0010	X	P0010	940
0700 24 0187 0285	0187 O OS	24 P0011	X	P0011	942
0174 69 1199 8003	O QL	69 SMAXM	8003		944

SUBROUTINE 1. INITIALIZE AT START OF A PROGRAM

HEO					946
					948
0006 24 0178 0231	0178 N SUBR1	24 EXITY		EXITY STORE EXIT	950
0231 61 0134 8003		61	8003	CLEAR REGION AND SYMBOL TABLES)	952
0134 20 189^ 0302	1899 J	20 ZMAX1		ZMAX1 CLEAR REGION AND SYMBOL TABLES)	954
0302 10 0405 0309		10 QA		CLEAR REGION AND SYMBOL TABLES)	956
0309 46 0112 0213		46	OUT	CLEAR REGION AND SYMBOL TABLES)	958
0112 11 0365 8003		11 QSA	8003	CLEAR REGION AND SYMBOL TABLES)	960
0213 20 0181 0234	0181 N OUT	20 P0005		P0005 CLEAR CARD NUMBER	962
0234 69 1044 0647		69 81STX		PRESET DRUM TAG)	964
0647 24 1669 0072	1669	24 DRUMT		DRUMT PRESET DRUM TAG)	966
0072 65 0175 0429		65 AVAL		MAKE DRUM AVAILABLE)	968
0429 10 0382 8003		10	8003	MAKE DRUM AVAILABLE)	970
0382 20 1554 1307	1554 J	20 A0200		A0200 MAKE DRUM AVAILABLE)	972
1307 11 0610 0415		11 QB		MAKE DRUM AVAILABLE)	974
0415 46 0178 0569		46 EXITY		MAKE DRUM AVAILABLE)	976
0569 10 0322 8003		10 QSB	8003	MAKE DRUM AVAILABLE)	978
0405 20 1661 0000	1661 Q QA	20 G0002	0000	G0002 COMPARISON CONSTANT	980
0365 20 1660 0000	1660 Q QSA	20 G0001	0000	G0001 RESTORE COMPARISON CONSTANT AND MODIFY	982
0610 20 1356 0000	1356 Q QB	20 A0002	0000	A0002 COMPARISON CONSTANT	984
0322 20 1355 0000	1355 Q QSB	20 A0001	0000	A0001 RESTORE COMPARISON CONSTANT AND MODIFY	986
0175 11 1111 1111	K AVAL	11 1111	1111	AVAILABILITY WORD	988

SUBROUTINE 2. RESERVE OR UNRESERVE A BLOCK OF DRUM

HEO					990
EQU SA					992
HSYMB					994
DOUBLE-PURPOSE TO CONSERVE DRUM SPACE					
0750 21 0304 0458	0304 R XX	21 UH	UH	REPLACE DIGIT OF AVAIL. WORD BY W)	998
0458 35 0001 0465		35 0001		REPLACE DIGIT OF AVAIL. WORD BY W)	1000
0465 65 8002 0723		65 8002		REPLACE DIGIT OF AVAIL. WORD BY W)	1002
0723 10 0326 0281		10 W		REPLACE DIGIT OF AVAIL. WORD BY W)	1004
0281 30 0001 0337		30 0001		REPLACE DIGIT OF AVAIL. WORD BY W)	1006
0337 10 0304 0359		10 UH	SPR	REPLACE DIGIT OF AVAIL. WORD BY W)	1008
0900 65 1003 0508	R 0900	65 N		HAVE WE FINISHED	1010
0508 45 0162 0178		45	EXITY	D, WE HAVE NOT FINISHED. I, WE HAVE.	1012
0162 16 0515 0619		16 110TH		REDUCE NUMBER YET TO BE DONE)	1014
0619 20 1003 0456	1003	20 N	N	REDUCE NUMBER YET TO BE DONE)	1016
0456 60 0108 0263		60 SA		ARE WE TO LAST LINE OF TABLE)	1018
0263 11 0116 0671		11 796		ARE WE TO LAST LINE OF TABLE)	1020
0671 46 0324 0225		46	TP	D, NOT LAST LINE.	1022
0324 10 0427 0331		10 800	SU	RESTORE, AND TAKE WORD FROM NEXT LINE	1024
0331 21 0108 0361	0108 C SU	21 SA	SA	STORE VARIABLE STORING ORDER	1026
0361 10 0264 8003		10 C4	8003	MAKE VARIABLE BRINGING ORDER	1028

1025 24 0178 0381	0178	N SUB2R	24 EXITY		EXITY	ENTRY TO RESERVE. STORE EXIT.	1030
0381 69 1680 0533			69 ZER0X	RU		BRING ZERO TO INDICATE RESERVATION	1032
1075 24 0178 0431	0178	N SUB2U	24 EXITY		EXITY	ENTRY TO UNRESERVE. STORE EXIT.	1034
0431 69 0515 0533			69 110TH	RU		BRING UNITY TO INDICATE UNRESERVATION	1036
0533 24 0326 0479	0326	C RU	24 W		W	STORE 0 OR 1	1038
0479 21 0284 0387	0284		21 XXXXA		XXXXA	STORE INITIAL ADDRESS TO BE RESERVED	1040
0387 20 1003 0506	1003		20 N		N	STORE N-1	1042
0506 11 0409 0313			11 2000I			IS INITIAL ADDRESS LESS THAN 2000	1044
0313 46 0166 0178			46	EXITY		1, OVER 1999. WE RESERVE NOTHING	1046
0166 10 1003 0558			10 N			UPPER, LAST ADDRESS - 2000	1048
0558 46 0411 0212			46 OK			1, LAST ONE WOULD BE OVER 1999	1050
0212 16 8003 0669			16 8003			MODIFY TO ENO WITH 1999)	1052
0669 16 0515 0719			16 110TH			MODIFY TO ENO WITH 1999)	1054
0719 20 1003 0411	1003		20 N	OK	N	MODIFY TO ENO WITH 1999)	1056
0411 60 0284 0389		C OK	60 XXXXA			MAKE SEVERAL VARIABLE ORDERS)	1058
0389 10 8001 0295			10 8001			MAKE SEVERAL VARIABLE ORDERS)	1060
0295 30 0003 1053			30 0003			MAKE SEVERAL VARIABLE ORDERS)	1062
1053 21 0658 0461	0658		21 X		X	MAKE SEVERAL VARIABLE ORDERS)	1064
0461 11 8001 0367			11 8001			MAKE SEVERAL VARIABLE ORDERS)	1066
0367 35 0001 0773			35 0001			MAKE SEVERAL VARIABLE ORDERS)	1068
0773 21 0228 0481	0228		21 P		P	MAKE SEVERAL VARIABLE ORDERS)	1070
0481 11 8001 0437			11 8001			MAKE SEVERAL VARIABLE ORDERS)	1072
0437 35 0002 0543			35 0002			MAKE SEVERAL VARIABLE ORDERS)	1074
0543 10 8003 0401			10 8003			MAKE SEVERAL VARIABLE ORDERS)	1076
0401 10 0658 0363			10 X			MAKE SEVERAL VARIABLE ORDERS)	1078
0363 15 0228 0583			15 P			MAKE SEVERAL VARIABLE ORDERS)	1080
0583 35 0004 0593			35 0004			MAKE SEVERAL VARIABLE ORDERS)	1082
0593 69 0246 0249			69 C1			MAKE SEVERAL VARIABLE ORDERS)	1084
0249 22 0901 0354	0901		22 0901		0901	MAKE SEVERAL VARIABLE ORDERS)	1086
0354 69 0708 0511			69 C2			MAKE SEVERAL VARIABLE ORDERS)	1088
0511 22 0359 0262	0359		22 SPR		SPR	MAKE SEVERAL VARIABLE ORDERS)	1090
0262 10 0565 0331			10 C3	SU		MAKE SEVERAL VARIABLE ORDERS)	1092
0225 65 0359 0413		N TP	65 SPR			BOTTOM LINE OF TABLE	1094
0413 16 0708 0463			16 C2			IS IT ALSO RIGHT ENO OF WORD	1096
0463 45 0216 0417			45	ZP		0, NO	1098
0216 15 0769 0973			15 C5			MODIFY TO TAKE NEXT COLUMN OF TABLE)	1100
0973 69 0246 0299			69 C1			MODIFY TO TAKE NEXT COLUMN OF TABLE)	1102
0299 22 0901 0404	0901		22 0901		0901	MODIFY TO TAKE NEXT COLUMN OF TABLE)	1104
0404 20 0359 0312	0359		20 SPR		SPR	MODIFY TO TAKE NEXT COLUMN OF TABLE)	1106
0312 60 0108 0513			60 SA			MODIFY TO TAKE FIRST LINE, SAME COLUMN	1108
0513 11 0266 0331			11 C6	SU		MODIFY TO TAKE FIRST LINE, SAME COLUMN	1110
0417 69 0246 0349		N ZP	69 C1			LAST LINE AND ENO OF WORD. MODIFY)	1112
0349 24 0901 0454	0901		24 0901		0901	LAST LINE AND ENO OF WORD. MODIFY)	1114
0454 69 0758 0561			69 C7			LAST LINE AND ENO OF WORD. MODIFY)	1116
0561 24 0359 0362	0359		24 SPR		SPR	LAST LINE AND ENO OF WORD. MODIFY)	1118
0362 60 0108 0563			60 SA			LAST LINE AND ENO OF WORD. MODIFY)	1120
0563 11 0316 0331			11 C8	SU		LAST LINE AND ENO OF WORD. MODIFY)	1122
0246 35 0000 0750		P C1	35 0000	XX		TO PRESET LEFT SHIFT ORDER	1124
0708 30 0009 0108		P C2	30 0009	SA		PRESETTER AND ALSO COMPARISON CONSTANT	1126
0565 20 1355 0900	1355	P C3	20 A0001	0900	A0001	INITIAL OF STORING ORDER	1128
0264 45 0000 0001		P C4	45 0000	0001		DIFFERENCE OF BRINGING AND STORING ORD	1130
0116 20 1551 0900	1551	Q 796	20 A0197	0900	A0197	COMPARISON CONSTANT FOR LAST LINE	1132
0427 20 1555 0900	1555	Q 800	20 A0201	0900	A0201	TO RESTORE FOR NEXT LINE OF TABLE	1134
0769 30 0010 0108		O C5	30 0010	SA		TO RESTORE AFTER SUBTRACTING C2	1136
0266 00 0196 0000		P C6	00 0196	0000		TO MODIFY BRINGING ORDER FOR NEXT COL	1138
0758 30 0000 0108		P C7	30 0000	SA		PRESETTER FOR START OF NEW COLUMN	1140
0316 00 0195 0000		P C8	00 0195	0000		MODIFY BRINGING ORDER TO START NEW COL	1142

SUBR 3 FINO WHETHER A00R IN UPPER IS FIXEO

HEO

0061 24 0177 0030	0177	N SUBR3	24 EXITX		EXITX	STORE EXIT	1148
0030 69 0633 0063			69	SUBR9		TO SUBROUTINE 9 TO ANALYZE THE ADDRESS	1150
0633 65 0536 0220		J	65 EQUIV	F0003		BRING EQUIV AND MULTIBRANCH	1152
0218 60 0068 0177		R F0001	60 810TH	EXITX		ADDRESS IS N G	1154
0219 65 0068 0177		R F0002	65 810TH	EXITX		BLANK ADDRESS	1156
0220 65 1680 0177		N F0003	65 ZER0X	EXITX		ORUM ADDRESS, OR EQUIVALENT TO ORUM	1158
0221 65 0068 0177		R F0004	65 810TH	EXITX		SYMBOLIC ADDRESS, WITH UNDEFINED SYMB	1160

0222 35 0005 0177	R F0005	35 0005	EXITX	800X ADDRESS	1162
0223 65 0068 0177	R F0006	65 810TH	EXITX	CDRE ADDRESS	1164
0224 60 0068 0177	R F0007	60 810TH	EXITX	OTHER ADDRESS. USUALLY PART OF CONST.	1166

					SUBR 4 PROCESS BACKWARDS L		1168
					HED		1170
0305 24 0177 0080	0177	N SUBR4	24 EXITX		EXITX STORE EXIT	1172	
0080 60 0151 0455			60 R0001		ALPHABETICAL L INTO UPPER	1174	
0455 69 0908 0063			69 PRE	SUBR9	ANALYZE THE ADDRESS	1176	
0908 65 0536 0271		J PRE	65 EDUIV	M0004	EQUIVALENT TO LOWER. MULTIBRANCH	1178	
0268 20 1679 0271	1679	N M0001	20 BLANB	M0004	BLANB STORE EQUIVALENT OF BLANK ADDRESS	1180	
0269 60 0622 0627		R M0002	60 87THX	SB10B	N. G. DMIT PUNCHING	1182	
0270 69 1044 0697		R M0003	69 81STX		BLANK. CHANGE FIRST-CARD TAG 1	1184	
0697 24 1050 1103	1050		24 1050		CHANGE FIRST-CARD TAG 1	1186	
1103 65 0556 0611			65	SUB11	FIND BEST EQUIVALENT TO BLANK	1188	
0556 00 0989 0268			00 0989	M0001	TAGS AND EXIT FOR SUBROUTINE 11	1190	
0271 20 018* 0335	0182	N M0004	20 P0006	SB10C	P0006 DRUM ADDRESS. STORE FOR PUNCHING	1192	
0272 65 0325 0611		R M0005	65	SUB11	NEW SYMBOL. FIND BEST EQUIVALENT	1194	
0325 00 0989 0276		J	00 0989	M0009	TAGS AND EXIT FOR USE IN SUBROUTINE 11	1196	
0273 01 0555 0273		R M0006	01 0555	M0006	800X ADDRESS. STDP THE MACHINE	1198	
0274 20 0182 0385	0182	R M0007	20 P0006		P0006 CORE ADDRESS. STORE FOR PUNCHING	1200	
0385 69 0538 0441			69	SUB13	FIND DYNAMIC LEVEL OF CDRE ADDRESS	1202	
0538 00 0909 0342		J	00 0909		TAGS AND EXIT FOR USE IN SUBR 13	1204	
0342 20 0391 0335	0391		20 ORCEB	SB10C	DRCEB STORE DYNAMIC LEVEL OF CORE ADDRESS	1206	
0275 01 0555 0275		R M0008	01 0555	M0008	OTHER ADDRESS. STDP MACHINE	1208	
0276 69 0197 0950		N M0009	69 ALOPT		EDUIV OF NEW SYMBOL.	1210	
0950 90 0271 0505			90 M0004		D. CASE WHERE WE DUPLICATE COMPUTATION	1212	
0505 69 0908 0661			69 PRE	SUBR7	USUAL CASE. STORE SYMBOL AND EQUIV	1214	
0277 00 0000 0269		R M0010	00 0000	M0002	NEW SYMBOL. BUT SYMBOL TABLE FULL	1216	

					SUBROUTINE 5 PROCESS BACKWARDS D		1218
					HED		1220
0400 24 0177 0230	0177	N SUBR5	24 EXITX		EXITX STORE EXIT	1222	
0230 60 0152 0958			60 R0002		ALPHABETIC D INTO UPPER	1224	
0958 69 0711 0063			69 PRE	SUBR9	ANALYZE THE ALPHABETIC D ADDRESS	1226	
0711 65 0536 0921		J PRE	65 EQUIV	C0004	BRING EQUIVALENT. MULTIBRANCH	1228	
0918 20 1023 0376	1023	N C0D01	20 BLANK	A	BLANK STORE EQUIV OF BLANK FOR FORWARD L	1230	
0919 60 0672 0627		R C0002	60 88THX	SB10B	N. G. OMIT PUNCHING	1232	
0920 69 1050 1353		R C0003	69 1050		BLANK ADDRESS. WAS THERE BLANK BACK L	1234	
1353 90 1008 1058			90	BB1	D, THERE WAS A BLANK BACKWARDS L	1236	
1008 65 1679 0376			65 BLANB	A	BRING EQUIVALENT OF BLANK BACKWARD L	1238	
1058 65 0761 0611		N BB1	65	SUB11	GET BLANK D DYNAMICALLY FROM I	1240	
0761 00 0988 0918		J	00 0988	C0001	TAGS AND EXIT FOR SUBROUTINE 11	1242	
0921 69 0624 0677		N C0004	69	INDEX	DRUM OR EQUIVALENT. INDEX IF TAGGED	1244	
0624 00 0000 0376		J	00 0000	A	TAG-IDENTIFICATION, AND EXIT FROM 18	1246	
0922 65 0625 0611		R C0005	65	SUB11	NEW SYMBOL. FIND BEST EQUIVALENT	1248	
0625 00 0988 0926		J	00 0988	C0009	EXIT AND TAGS FOR SUBROUTINE 11	1250	
0923 69 0426 0441		R C0006	69	SUB13	800X ADDRESS. GET DYNAMIC LEVEL	1252	
0426 00 0908 0280		J	00 0908		TAGS AND EXIT FOR SUBROUTINE 13	1254	
0280 69 0683 0586			69	SUB17	CORRECTION TO DYNAMIC LEVEL	1256	
0683 66 8002 0491		J	66 8002		CHANGE SIGN OF CORRECTION	1258	
0491 15 0294 0335			15 XXXX2	SB10C	ADD TO GIVE MODIFIED DYNAMIC LEFEL	1260	
0924 69 0727 0677		R C0007	69	INDEX	CORE ADDRESS. INDEX IT	1262	

0727 00 0000 0631	J	00 0000		ADDEND AND EXIT FOR SUBROUTINE 18	1264
0631 20 0184 0487		20 P0008	P0008	STORE CORE ADDRESS FOR PUNCHING	1266
0487 69 0190 0441	0184	69	SUB13	GET DYNAMIC LEVEL OF CORE ADDRESS	1268
0190 00 0908 0344	J	00 0908		TAGS AND EXIT FOR SUBROUTINE 13	1270
0344 69 1050 0504		69 1050		FIRST-CARD TAG	1272
0504 90 0335 0459		90 SB10C		D, WE SHOULD STORE THIS DYNAMIC LEVEL	1274
0459 20 0486 0335	0486	20 SAVOR	SB10C	SAVOR STORE DYN LEV OF CORE, AND FINISH UP	1276
0376 20 0062 0925	0062	20 OPREG	C0008	OPREG STORE ADDRESS FOR OPTIMIZING NEXT ADDR	1278
0925 20 0184 0177	0184	20 P0008	EXITX	P0008 MISC ADDR. STORE FOR PUNCHING. EXIT	1280
0926 69 0711 0661	N C0009	69 PRE	SUBR7	STORE EQUIVALENT OF NEW SYMBOL	1282
0927 00 0000 0919	R C0010	00 0000	C0002	NEW SYMBOL, BUT TABLE FULL. N.G.	1284

SUBROUTINE 6 PROCESS BACKWARDS I					1286
HED					1288
0142 24 0177 0330	0177	N SUBR6	24 EXITX	EXITX STORE EXIT	1290
0330 60 0153 1108			60 R0003	ALPHABETIC I TO UPPER	1292
1108 69 0911 0063			69 PRE	ANALYZE ALPHABETIC I	1294
0911 65 0536 0371	J PRE	65 EQUIV	J0004	BRING EQUIVALENT. MULTIBRANCH	1296
0368 69 0911 0661	N J0001	69 PRE	SUBR7	STORE EQUIVALENT OF NEW SYMBOL	1298
0369 60 0722 0627	R J0002	60 89THX	SB10B	N.G. OMIT PUNCHING	1300
0370 69 1669 0772	R J0003	69 DRUMT		BLANK I, IS DRUM FULL	1302
0772 90 0626 0369		90	J0002	D, DRUM IS NOT FULL	1304
0626 65 1679 0371		65 BLANB	J0004	BRING EQUIVALENT OF BLANK BACKWARD L	1306
0371 24 0062 0615	0062	24 OPREG		DRUM OR DRUM EQUIVALENT	1308
0615 69 0568 0677		69	INDEX	TO SUBROUTINE 18 TO INDEX	1310
0568 00 0001 0375	J	00 0001	J0008	ADDEND AND EXIT FOR SUBROUTINE 18	1312
0372 65 0675 0611	R J0005	65	SUB11	NEW SYMBOL. FIND BEST VALUE	1314
0675 00 0990 0368	J	00 0990	J0001	TAGS AND EXIT FOR SUBROUTINE 11	1316
0373 01 0000 0373	R J0006	01 0000	J0006	800X RANGE. ERROR.	1318
0374 69 0391 0371	R J0007	69 ORCEB	J0004	CORE. BRING DYNAMIC LEVEL	1320
0375 20 0185 0177	0185	R J0008	20 P0009	EXITX P0009 OTHER ADDRESS, USUALLY A CONSTANT	1322

SUBROUTINE 7 STORE SYMBOL AND ITS EQUIVALENT					1324
HED					1326
0661 24 0179 0432	0179	N SUBR7	24 EXITZ	EXITZ STORE EXIT	1328
0432 65 0435 0439			65 LSymb	SYMBOL-STORING ORDER)	1330
0439 15 0392 0747			15 SS	SYMBOL-STORING ORDER)	1332
0747 69 0108 8002			69 HSYMB	8002 STORE SYMBOL)	1334
0392 24 1689 0442	1689	J SS	24 Z0001	Z0001 STORE SYMBOL)	1336
0442 65 0435 0489			65 LSymb	LOCATION OF EQUIVALENT RELATIVE TO E1	1338
0489 14 0492 1029			14 2DXXX	LOCATION OF EQUIVALENT RELATIVE TO E1	1340
1029 35 0004 1039			35 0004	LOCATION OF EQUIVALENT RELATIVE TO E1	1342
1039 20 0294 0797	0294		20 XXXX2	LOCATION OF EQUIVALENT RELATIVE TO E1	1344
0797 15 1000 0555			15 P1	MAKE BRINGING ORDER)	1346
0555 20 0167 0670	0167		20 XXXX1	MAKE BRINGING ORDER)	1348
0670 44 1073 0674			44 IPOS	D, USE I-POSITION. I, USE D-POSITION.	1350
0674 65 0536 0541			65 EQUIV	BRING EQUIVALENT	1352
0541 35 0004 0451			35 0004	SHIFT TO D POSITION	1354
0451 10 0554 0509			10 P2	MAKE STORING ORDER)	1356
0509 10 0294 0167	C A		10 XXXX2	MAKE STORING ORDER)	1358
1073 65 0536 0591	N IPOS	65 EQUIV		USE I-POSITION. BRING EQUIVALENT	1360
0591 10 0394 0509		10 P3	A	MAKE STORING ORDER	1362
1000 69 1555 8003	P P1	69 E0001	8003	INITIAL OF BRINGING ORDER	1364
0554 22 1555 0179	1555	P P2	22 E0001	EXITZ E0001 INITIAL OF ORDER TO STORE D-POSITION	1366
0394 23 1555 0179	1555	P P3	23 E0001	EXITZ E0001 INITIAL OF ORDER TO STORE I-POSITION	1368

SUBR 8 PROCESS TYPE 2 ADDR. ALPH IN UPPER, NUMERIC, LOWR					1370
HED					1372
1024 24 0177 0380	0177	N SUBR8	24 EXITX	EXITX STORE EXIT	1374

0380 11 0254 0559		11 11STX		IS FIRST POSITION BLANK	1376
0559 46 0412 0613		46	FXT	I, FIRST IS NON-BLANK. FIXED ADDRESS.	1378
0412 10 8001 0969		10 8001		RESTORE TO POSITIVE	1380
0969 60 8002 0777		60 8002		NUMERIC ADDRESS ALONE IN UPPER	1382
0777 11 0409 0663		11 2000I		SUBTRACT 2000	1384
0663 46 0366 0467		46	RC	O, DRUM. I, CORE.	1386
0366 10 1019 1123		10 00IFF		DRUM. ADD RELOCATION AMOUNT	1388
1123 46 0676 0977		46	BAO	I, RELOCATED DRUM WILL EXCEED 1999	1390
0676 10 0409 0713		10 2000I	RES	RESTORE RELOCATED DRUM ADDRESS	1392
0467 10 8001 1223	N RC	10 8001		CORE ADDRESS. RESTORE THE 2000	1394
1223 10 0726 0681		10 00IFF		ADD RELOCATION AMOUNT	1396
0681 11 0334 0539		11 9060		SUBTRACT 9060	1398
0539 46 0542 0977		46	BAO	I, RELOCATED CORE ADDRESS TOO HIGH	1400
0542 10 8001 0713		10 8001	RES	RESTORE THE 9060	1402
0713 21 0167 0720	0167 C RES	21 XXXX1	XXXX1	STORE ADDRESS TEMPORARILY	1404
0720 69 1273 1025		69	SUB2R	RESERVE IF DRUM ADDRESS	1406
1273 65 0167 0177	J	65 XXXX1	EXITX	EQUIVALENT BACK TO LOWER AND EXIT	1408
0613 65 8002 0177	N FXT	65 8002	EXITX	FIXED ADDRESS. CLEAR UPPER AND EXIT.	1410
0977 60 0622 0177	N BAD	60 87THX	EXITX	MODIFIED ADDR TOO HIGH. 8000 TO UPPER	1412
0334 00 0000 9060	K 9060	00 0000	9060		1414

SUBROUTINE 9. ANALYZE AN ADDRESS

HEO

1002 44 1005 0606	C LOOP	44	UND	I, SYMBOL IS NOT IN THE TABLE	1416
1005 11 0108 0763		11 HSYMB		SOME SYMBOL WAS FOUND. SUBTRACT OURS	1418
0763 44 0517 0618		44 NZ		O, IT WAS NOT RIGHT ONE	1420
0618 46 0517 0972		46 NZ	DEF	O, IT WAS NOT RIGHT ONE. I, IT AS.	1422
0517 10 8001 1323	C NZ	10 8001		WRONG SYMBOL. RESTORE UPPER TO PLUS	1424
1323 65 8002 0731		65 8002		CLEAR UPPER	1426
0731 15 0384 8002		15 T	8002	ADVANCE LOWER AND BRING ANOTHER	1428
0063 24 0179 0482	0179 N SUBR9	24 EXITZ		EXITZ STORE EXIT	1430
0482 45 0636 0537		45 ABC		I, THE ADDRESS IS BLANK	1432
0537 61 0515 1069		61 110TH	00N	MODIFY EXIT AND PREPARE TO QUIT	1434
0636 21 0108 0961	0108 N ABC	21 HSYMB		HSYMB ADDRESS NOT BLANK. STORE SYMBOL	1436
0961 30 0008 0629		30 0008		ALSO STORE FIRST LETTER OF ADDRESS)	1440
0629 21 0294 0947	0294	21 XXXX2	XXXX2	ALSO STORE FIRST LETTER OF ADDRESS)	1442
0947 11 8001 0604		11 8001		CLEAR FIRST LETTER	1444
0604 16 1208 0913		16 909		SUBTRACT 9090908995	1446
0913 46 0416 0567		46 SYM	LOO	O, ADDRESS IS SYMBOLIC.	1448
0567 11 8003 0725	C LOO	11 8003		ARE LAST FOUR CHARACTERS DIGITS)	1450
0725 45 0278 0679		45	OUT	ARE LAST FOUR CHARACTERS DIGITS)	1452
0278 35 0001 0485		35 0001		ARE LAST FOUR CHARACTERS DIGITS)	1454
0485 44 0416 0240		44 SYM		ARE LAST FOUR CHARACTERS DIGITS)	1456
0240 10 8001 0997		10 8001		ARE LAST FOUR CHARACTERS DIGITS)	1458
0997 35 0001 0567		35 0001	LOO	ARE LAST FOUR CHARACTERS DIGITS)	1460
0679 65 8001 0535	N OUT	65 8001		THEY ARE DIGITS. GET VALUE)	1462
0535 30 0001 0641		30 0001		THEY ARE DIGITS. GET VALUE)	1464
0641 20 0536 0589	0536	20 EQUIV	EQUIV	STORE THESE 4 DIGITS	1466
0589 10 0294 0399		10 XXXX2		FIRST CHARACTER INTO UPPER	1468
0399 44 0654 0704		44	ABS	I, ZERO. ADDRESS WAS ABSOLUTE	1470
0654 11 1308 0963		11 99THX		NON-ZERO. SUBTRACT 90	1472
0963 46 0466 0416		46	SYM	I, FIRST CHAR IS DIGIT. ADDR SYMBOLIC	1474
0466 10 1119 0724		10 29IXX		ADD 29	1476
0724 46 0416 0328		46 SYM		O, FIRST IS SPEC. CHAR. SYMBOLIC	1478
0328 35 0004 0639		35 0004		MAKE BRINGING ORDER)	1480
0639 10 0592 8003		10	8003	MAKE BRINGING ORDER)	1482
0592 65 1660 0665	J	65 00001		BRING EQUIVALENT OF REGION	1484
0665 45 0668 1219		45	NG	I, REGION IS UNDEFINED	1486
0668 15 0536 0691		15 EQUIV		GET EQUIVALENT OF REGIONAL ADDRESS)	1488
0691 16 0915 1269		16 110TH		GET EQUIVALENT OF REGIONAL ADDRESS)	1490
1269 23 0336 0704	0536	23 EQUIV	ABS	STORE EQUIVALENT	1492
0416 60 0108 1013	N SYM	60 HSYMB		SYMBOLIC ADDRESS. BRING SYMBOL	1494
1013 35 0008 0781		35 0008		CLEAR ALL BUT LAST CHARACTER	1496
0781 44 0585 0686		44	SHRT	O, LONG SYMBOL	1498
0585 69 0038 0741		69 16THX		PRESET T AS POSITIVE 1)	1500
0741 24 0384 0587	0384	24 T	T	PRESET T AS POSITIVE 1)	1502

0727	00	0000	0631		J	00	0000		ADDEND AND EXIT FOR SUBROUTINE 18	1264
0631	20	0184	0487	0184		20	P0008	P0008	STORE CORE ADDRESS FOR PUNCHING	1266
0487	69	0190	0441			69		SUB13	GET DYNAMIC LEVEL OF CORE ADDRESS	1268
0190	00	0908	0344		J	00	0908		TAGS AND EXIT FOR SUBROUTINE 13	1270
0344	69	1050	0504			69	1050		FIRST-CARD TAG	1272
0504	90	0335	0459			90	SB10C		D, WE SHOULD STORE THIS DYNAMIC LEVEL	1274
0459	20	0486	0335	0486		20	SAVOR	SB10C	SAVOR STORE DYN LEV OF CORE, AND FINISH UP	1276
0376	20	0062	0925	0062	N A	20	OPREG	C0008	OPREG STORE ADDRESS FOR OPTIMIZING NEXT ADDR	1278
0925	20	0184	0177	0184	R C0008	20	P0008	EXITX	P0008 MISC ADDR. STORE FOR PUNCHING. EXIT	1280
0926	69	0711	0661		N C0009	69	PRE	SUBR7	STORE EQUIVALENT OF NEW SYMBOL	1282
0927	00	0000	0919		R C0010	00	0000	C0002	NEW SYMBOL, BUT TABLE FULL. N.G.	1284

SUBROUTINE 6 PROCESS BACKWARDS I

HED

0142	24	0177	0330	0177	N SUBR6	24	EXITX		EXITX STORE EXIT	1290
0330	60	0153	1108			60	R0003		ALPHABETIC I TO UPPER	1292
1108	69	0911	0063			69	PRE	SUBR9	ANALYZE ALPHABETIC I	1294
0911	65	0536	0371		J PRE	65	EQUIV	J0004	BRING EQUIVALENT. MULTIBRANCH	1296
0368	69	0911	0661		N J0001	69	PRE	SUBR7	STORE EQUIVALENT OF NEW SYMBOL	1298
0369	60	0722	0627		R J0002	60	89THX	SB10B	N.G. OMIT PUNCHING	1300
0370	69	1669	0772		R J0003	69	DRUMT		BLANK I, IS DRUM FULL	1302
0772	90	0626	0369			90		J0002	D, DRUM IS NOT FULL	1304
0626	65	1679	0371	0062	N J0004	65	BLANB	J0004	BRING EQUIVALENT OF BLANK BACKWARD L	1306
0371	24	0062	0615			24	OPREG		DRUM OR DRUM EQUIVALENT	1308
0615	69	0568	0677			69		INDEX	TO SUBROUTINE 18 TO INDEX	1310
0568	00	0001	0375		J	00	0001	J0008	ADDEND AND EXIT FOR SUBROUTINE 18	1312
0372	65	0675	0611		R J0005	65		SUB11	NEW SYMBOL. FIND BEST VALUE	1314
0675	00	0990	0368		J	00	0990	J0001	TAGS AND EXIT FOR SUBROUTINE 11	1316
0373	01	0000	0373		R J0006	01	0000	J0006	800X RANGE. ERROR.	1318
0374	69	0391	0371		R J0007	69	ORCEB	J0004	CORE. BRING DYNAMIC LEVEL	1320
0375	20	0185	0177	0185	R J0008	20	P0009	EXITX	P0009 OTHER ADDRESS, USUALLY A CONSTANT	1322

SUBROUTINE 7 STORE SYMBOL AND ITS EQUIVALENT

HED

0661	24	0179	0432	0179	N SUBR7	24	EXITZ		EXITZ STORE EXIT	1328
0432	65	0435	0439			65	LSYMB		SYMBOL-STORING ORDER)	1330
0439	15	0392	0747			15	SS		SYMBOL-STORING ORDER)	1332
0747	69	0108	8002			69	HSYMB	8002	STORE SYMBOL)	1334
0392	24	1689	0442	1689	J SS	24	Z0001		Z0001 STORE SYMBOL)	1336
0442	65	0435	0489			65	LSYMB		LOCATION OF EQUIVALENT RELATIVE TO E1	1338
0489	14	0492	1029			14	2DXXX		LOCATION OF EQUIVALENT RELATIVE TO E1	1340
1029	35	0004	1039			35	0004		LOCATION OF EQUIVALENT RELATIVE TO E1	1342
1039	20	0294	0797	0294		20	XXXX2		LOCATION OF EQUIVALENT RELATIVE TO E1	1344
0797	15	1000	0555			15	P1		MAKE BRINGING ORDER)	1346
0555	20	0167	0670	0167		20	XXXX1		MAKE BRINGING ORDER)	1348
0670	44	1073	0674			44	IPOS		D, USE I-POSITION. I, USE D-POSITION.	1350
0674	65	0536	0541			65	EQUIV		BRING EQUIVALENT	1352
0541	35	0004	0451			35	0004		SHIFT TO D POSITION	1354
0451	10	0554	0509			10	P2	A	MAKE STORING ORDER)	1356
0509	10	0294	0167		C A	10	XXXX2	XXXX1	MAKE STORING ORDER)	1358
1073	65	0536	0591		N IPOS	65	EQUIV		USE I-POSITION. BRING EQUIVALENT	1360
0591	10	0394	0509			10	P3	A	MAKE STORING ORDER	1362
1000	69	1555	8003		P P1	69	E0001	8003	INITIAL OF BRINGING ORDER	1364
0554	22	1555	0179	1555	P P2	22	E0001	EXITZ	E0001 INITIAL OF ORDER TO STORE D-POSITION	1366
0394	23	1555	0179	1555	P P3	23	E0001	EXITZ	E0001 INITIAL OF ORDER TO STORE I-POSITION	1368

SUBR 8 PROCESS TYPE 2 ADDR. ALPH IN UPPER, NUMERIC, LOWR

HED

1024	24	0177	0380	0177	N SUBR8	24	EXITX		EXITX STORE EXIT	1374
------	----	------	------	------	---------	----	-------	--	------------------	------

0380 11 0254 0559		11 11STX		IS FIRST POSITION BLANK	1376
0559 46 0412 0613		46	FXT	I, FIRST IS NON-BLANK. FIXED ADDRESS.	1378
0412 10 8001 0969		10 8001		RESTORE TO POSITIVE	1380
0969 60 8002 0777		60 8002		NUMERIC ADDRESS ALONE IN UPPER	1382
0777 11 0409 0663		11 2000I		SUBTRACT 2000	1384
0663 46 0366 0467		46	RC	O, DRUM. I, CORE.	1386
0366 10 1019 1123		10 00IFF		DRUM. ADD RELOCATION AMOUNT	1388
1123 46 0676 0977		46	8A0	I, RELOCATED DRUM WILL EXCEED 1999	1390
0676 10 0409 0713		10 2000I	RES	RESTORE RELOCATED DRUM ADDRESS	1392
0467 10 8001 1223	N RC	10 8001		CORE ADDRESS. RESTORE THE 2000	1394
1223 10 0726 0681		10 00IFF		ADD RELOCATION AMOUNT	1396
0681 11 0334 0539		11 9060		SUBTRACT 9060	1398
0539 46 0542 0977		46	8A0	I, RELOCATED CORE ADDRESS TOO HIGH	1400
0542 10 8001 0713		10 8001	RES	RESTORE THE 9060	1402
0713 21 0167 0720	0167 C RES	21 XXXX1	XXXX1	STORE ADDRESS TEMPORARILY	1404
0720 69 1273 1025		69	SUB2R	RESERVE IF DRUM ADDRESS	1406
1273 65 0167 0177	J	65 XXXX1	EXITX	EQUIVALENT BACK TO LOWER AND EXIT	1408
0613 65 8002 0177	N FXT	65 8002	EXITX	FIXED ADDRESS. CLEAR UPPER AND EXIT.	1410
0977 60 0622 0177	N BAO	60 87THX	EXITX	MODIFIED ADDR TOO HIGH. 8000 TO UPPER	1412
0334 00 0000 9060	K 9060	00 0000	9060		1414
SUBROUTINE 9. ANALYZE AN ADDRESS					
	HE0				1416
1002 44 1005 0606	C LOOP	44	UND	I, SYMBOL IS NOT IN THE TABLE	1420
1005 11 0108 0763		11 HSYMB		SOME SYMBOL WAS FOUND. SUBTRACT OURS	1422
0763 44 0517 0618		44 NZ		O, IT WAS NOT RIGHT ONE	1424
0618 46 0517 0972		46 NZ	DEF	O, IT WAS NOT RIGHT ONE. I, IT AS.	1426
0517 10 8001 1323	C NZ	10 8001		WRONG SYMBOL. RESTORE UPPER TO PLUS	1428
1323 65 8002 0731		65 8002		CLEAR UPPER	1430
0731 15 0384 8002		15 T	8002	ADVANCE LOWER AND BRING ANOTHER	1432
0063 24 0179 0482	0179 N SUBR9	24 EXITZ		EXITZ STORE EXIT	1434
0482 45 0636 0537		45 ABC		I, THE ADDRESS IS BLANK	1436
0537 61 0515 1069		61 110TH	00N	MODIFY EXIT AND PREPARE TO OUIT	1438
0636 21 0108 0961	0108 N ABC	21 HSYMB	HSYMB	ADDRESS NOT BLANK. STORE SYMBOL	1440
0961 30 0008 0629		30 0008		ALSO STORE FIRST LETTER OF ADDRESS)	1442
0629 21 0294 0947	0294	21 XXXX2	XXXX2	ALSO STORE FIRST LETTER OF ADDRESS)	1444
0947 11 8001 0604		11 8001		CLEAR FIRST LETTER	1446
0604 16 1208 0913		16 909		SUBTRACT 9090908995	1448
0913 46 0416 0567		46 SYM	LOO	O, ADDRESS IS SYMBOLIC.	1450
0567 11 8003 0725	C LOO	11 8003		ARE LAST FOUR CHARACTERS DIGITS)	1452
0725 45 0278 0679		45	OUT	ARE LAST FOUR CHARACTERS DIGITS)	1454
0278 35 0001 0485		35 0001		ARE LAST FOUR CHARACTERS DIGITS)	1456
0485 44 0416 0240		44 SYM		ARE LAST FOUR CHARACTERS DIGITS)	1458
0240 10 8001 0997		10 8001		ARE LAST FOUR CHARACTERS DIGITS)	1460
0997 35 0001 0567		35 0001	LOO	ARE LAST FOUR CHARACTERS DIGITS)	1462
0679 65 8001 0535	N OUT	65 8001		THEY ARE DIGITS. GET VALUE)	1464
0535 30 0001 0641		30 0001		THEY ARE DIGITS. GET VALUE)	1466
0641 20 0536 0589	0536	20 EQUIV	EQUIV	STORE THESE 4 DIGITS	1468
0589 10 0294 0399		10 XXXX2		FIRST CHARACTER INTO UPPER	1470
0399 44 0654 0704		44	ABS	I, ZERO. ADDRESS WAS ABSOLUTE	1472
0654 11 1308 0963		11 99THX		NON-ZERO. SUBTRACT 90	1474
0963 46 0466 0416		46	SYM	I, FIRST CHAR IS DIGIT. ADDR SYMBOLIC	1476
0466 10 1119 0724		10 29IXX		ADD 29	1478
0724 46 0416 0328		46 SYM		O, FIRST IS SPEC. CHAR. SYMBOLIC	1480
0328 35 0004 0639		35 0004		MAKE BRINGING ORDER)	1482
0639 10 0592 8003		10	8003	MAKE BRINGING ORDER)	1484
0592 65 1660 0665	J	65 G0001		BRING EQUIVALENT OF REGION	1486
0665 45 0668 1219		45	NG	I, REGION IS UNDEFINED	1488
0668 15 0536 0691		15 EQUIV		GET EQUIVALENT OF REGIONAL ADDRESS)	1490
0691 16 0915 1269		16 110TH		GET EQUIVALENT OF REGIONAL ADDRESS)	1492
1269 23 0936 0704	0536	23 EQUIV	ABS EQUIV	STORE EQUIVALENT	1494
0416 60 0108 1013	N SYM	60 HSYMB		SYMBOLIC ADDRESS. BRING SYMBOL	1496
1013 35 0008 0781		35 0008		CLEAR ALL BUT LAST CHARACTER	1498
0781 44 0585 0686		44	SHRT	O, LONG SYMBOL	1500
0585 69 0038 0741		69 16THX		PRESET T AS POSITIVE 1)	1502
0741 24 0384 0587	0384	24 T	T	PRESET T AS POSITIVE 1)	1504

0587 65 0290 8001		65 Y	8001	BRINGING ORDER INTO LOWER	1506
0290 10 1689 1002	J Y	10 Z0001	LOOP	ADD A SYMBOL INTO UPPER, FROM BOTTOM	1508
0686 66 0038 0643	N SHRT	66 16THX		SHORT, PRESET T AS -1)	1510
0643 20 0384 0637	0384	20 T	T	SHORT, PRESET T AS -1)	1512
0637 65 0340 8001		65	8001	BRINGING ORDER INTO LOWER	1514
0340 10 1898 1002	J	10 ZMAXM	LOOP	ADD A SYMBOL, STARTING AT TOP	1516
0606 16 0290 0345	N UND	16 Y		SYMB NOT IN TABLE, SUBTRACT Y	1518
0345 20 0435 0588	0435	20 LSymb	LSymb	STORE ADDR OF SPACE RELATIVE TO START	1520
0588 46 0791 0642		46 FULL		0, TABLE IS FULL	1522
0642 16 0395 0449		16 ZTABL			1524
0449 46 0352 0791		46	FULL	1, TABLE IS FULL	1526
0352 60 0515 1069		60 110TH	DON	MODIFY EXIT, AND PREPARE TO QUIT	1528
0972 16 0290 0445	N DEF	16 Y		DEFINED SYMBOL, SUBTRACT Y	1530
0445 16 0498 0754		16 JBRL		SUBTRACT INITIAL BRINGING ORDER	1532
0754 20 0435 0638	0435	20 LSymb	LSymb	ADDRESS OF SYMBOL RELATIVE TO START	1534
0638 14 0492 0904		14 2DXXX		DIVIDE BY 2	1536
0904 35 0004 0715		35 0004		QUOTIENT INTO D OF LOWER	1538
0715 15 0718 8002		15	8002	MAKE BRINGING ORDER	1540
0718 69 1555 0659	J	69 E0001		BRING WORD SHOWING EQUIVALENT	1542
0659 44 1063 0314		44	LH	0, ADDR IS ODD. EQUIV IN RIGHT END	1544
1063 67 8001 1319		67 8001	LR	EQUIVALENT IS IN I POSITION OF LOWER	1546
0314 67 8001 0721	N LH	67 8001		EQUIVALENT IN 0 POSITION OF LOWER	1548
0721 30 0004 1319		30 0004	LR	EQUIVALENT IN I POSITION OF LOWER	1550
1319 69 8003 0776	C LR	69 8003		CLEAR DISTRIBUTOR	1552
0776 23 0536 0704	0536	23 EQUIV	ABS	EQUIV STORE EQUIVALENT OF SYMBOL	1554
0791 01 0111 1219	J FULL	01 0111	NG	STOP IF SYMBOL TABLE IS FULL	1556
1219 61 1022 1069	C NG	61 21XXX	DON	MODIFY EXIT FOR NG SYMBOL	1558
0704 35 0006 0770	N ABS	35 0006		ABSOLUTE ADDRESS OR EQUIVALENT	1560
0770 16 8002 0729		16 8002		ADDRESS INTO LEFT END DISTRIBUTOR	1562
0729 84 1200 0775		84 1200		SEARCH TABLE ACCORDING TO SIZE OF ADDR	1564
0775 15 0378 8002		15	8002	MAKE BRINGING ORDER	1566
0378 60 0000 0605		60 0000		BRING APPROPRIATE TABULAR VALUE	1568
0605 35 0006 0970		35 0006		SHIFT TO REMOVE TABULAR ADDRESS)	1570
0970 30 0006 1069		30 0006	DON	SHIFT TO REMOVE TABULAR ADDRESS)	1572
1069 10 0179 8003	C DON	10 EXITZ	8003	FROM MANY PLACES. MULTIBRANCH	1574
1208 90 9090 8995	K 909	90 9090	8995		1576

SUBROUTINE 10. MISCELLANEOUS EXITS

0001 60 0003 0709	N SETCC	60 READC	SUB10	SEQUEL TO MANY PSEUDO-OPS.	1580
0709 15 1044 0499	C SUB10	15 81STX		C.I. TO PUNCH X-9)	1582
0499 20 0186 0689	0186	20 P0010	P0010	C.I. TO PUNCH X-9)	1584
0689 60 8003 0057		60 8003	SB10A	CLEAR LOWER	1586
0057 15 0181 0635	C SB10A	15 P0005		ADVANCE CARD NUMBER)	1588
0635 15 1022 1027		15 21XXX		ADVANCE CARD NUMBER)	1590
1027 20 0181 0434	0181	20 P0005	P0005	ADVANCE CARD NUMBER)	1592
0434 71 0177 8003		71 P0001	8003	PUNCH CARD, PERFORM UPPER.	1594
0627 10 0186 0941	N SB10B	10 P0010		FROM 4-6,19-21. TO SUPPRESS)	1596
0941 21 0154 0759	0154	21 R0004		R0004 TO SUPPRESS IN CASE WE SEARCH	1598
0759 21 0186 0177	0186	21 P0010	EXITX	P0010 FROM 4-6,19-21. TO SUPPRESS)	1600
0335 20 0062 0177	0062	20 OPREG	EXITX	OPREG FROM 4-5, 19-20. STORE DYNAMIC LEVEL	1602

SUBROUTINE 11. FIND BEST LOCATION AND RESERVE IT

		HED			1604
					1606
0611 69 1669 1072	N SUB11	69 DRUMT		IS THE DRUM FULL	1608
1072 90 0976 1077		90	PAKT	I, YES.	1610
0976 20 0178 0931	0178	20 EXITY	EXITY	STORE EXIT OUT OF LOWER	1612
0931 96 0484 0736		96 DI		I, ADDRESS IS BACKWARD I OR FORWARD L	1614
0736 65 8003 0693		65 8003	SSW	CLEAR ACCUMULATOR AND JUMP	1616
0484 69 0687 0390	N DI	69 F		PREPARE EXIT AND GO TO SUBROUTINE 13)	1618
0390 22 0167 0441	0167	22 XXXX1	SUB13	PREPARE EXIT AND GO TO SUBROUTINE 13)	1620
0687 00 0000 0991	J F	00 0000		TO BUILD EXIT FROM SUBROUTINE 13	1622
0991 35 0004 0501		35 0004		4 TIMES DYNAMIC ADDRESS IN 0 POSITION	1624
0501 15 8002 0909		15 8002		4 TIMES DYNAMIC ADDRESS IN 0 POSITION	1626
0909 15 8002 0693		15 8002	SSW	4 TIMES DYNAMIC ADDRESS IN D POSITION	1628
0693 69 0178 0981	C SSW	69 EXITY		ALL CASES. GOING WHICH DIRECTION	1630

0981 97 0534 0786		97	BAK		0, FORWARD. I, BACKWARD.	1632
0534 69 0737 0440		69 OFF			FORWARD. SET SWITCH OFF FOR 1ST PART	1634
0440 24 0743 0296	0743	24 SW	SW		FORWARD. SET SWITCH OFF FOR 1ST PART	1636
0296 15 0549 0954		15 601			INITIAL OF VARIABLE BRINGING ORDER	1638
0954 20 0959 0462	0959	20 BP1	BP1		STORE VARIABLE BRINGING ORDER	1640
0462 16 0038 0793		16 16THX	ST		COMPARISON CONSTANT FOR LAST PART	1642
1088 44 1241 0743		44 YES	SW		IS A LOCATION AVAILABLE IN THIS GROUP	1644
0743 65 0346 0551		65 AI	SWOF		NOT PERFORMED—JUST FOR OPTIMIZING	1646
0737 65 0346 0551		65 AI	SWOF		HAVE WE REACHED TOP OF TABLE	1648
0551 16 1054 1009		16 799			SUBTRACT END OF TABLE	1650
1009 45 0512 1113		45	MAX		I, WE HAVE REACHED TOP OF TABLE	1652
0512 15 0765 1020		15 800	SAI		ADVANCE VARIABLE BRINGING ORDER	1654
1020 20 0346 8001	0346	20 AI	8001 AI		STORE BRINGING ORDER, AND DO IT	1656
0786 69 0739 0692		69 OFB			GOING BACKWARD. SET SWITCH OFF)	1658
0692 24 0743 0396	0743	24 SW	SW		GOING BACKWARD. SET SWITCH OFF)	1660
0396 15 0599 1104		15 602			INITIAL BRINGING ORDER)	1662
1104 20 0959 0562	0959	20 BP1	BP1		INITIAL BRINGING ORDER)	1664
0562 15 0038 0793		15 16THX	ST		COMPARISON CONSTANT FOR LAST PART	1666
0793 20 1097 1020	1097	20 A0	SAI A0		STORE COMPARISON CONSTANT	1668
0739 65 0346 0601		65 AI			START BACKWARD SEARCH. HOW FAR	1670
0601 16 1354 1059		16 600			SUBTRACT COMPARISON CONSTANT	1672
1059 45 0612 1213		45	MIN		0, WE ARE NOT BACK TO START YET	1674
0612 15 0915 1020		15 599	SAI		RESTORE AND MODIFY	1676
1113 69 0516 1070		69 ON			AT TOP. SET SWITCH ON)	1678
1070 24 0743 0446	0743	24 SW	SW		AT TOP. SET SWITCH ON)	1680
0446 65 1354 1020		65 600	SAI		RESET BRINGING ORDER, AND SEARCH MORE	1682
0516 65 0346 0651		65 AI			AFTER RESTART. BRING BRINGING ORDER	1684
0651 16 1097 0701		16 A0			COMPARISON CONSTANT, SUBTRACTED	1686
0701 45 0655 0705		45	FULL		I, THERE IS NO AVAILABLE LOCATION.	1688
0655 15 0959 1020		15 BP1	SAI		RESTORE AND MODIFY	1690
1213 69 0516 1120		69 ON			SECOND PART, BACKWARD. CHANGE SWITCH	1692
1120 24 0743 0496	0743	24 SW	SW		SECOND PART, BACKWARD. CHANGE SWITCH	1694
0496 65 1054 1020		65 799	SAI		START BACK FROM TOP OF TABLE	1696
0705 01 0222 1109		01 0222			STOP BECAUSE ORUM IS PACKED	1698
1109 69 0114 0617		69 91STX			CHANGE ORUM TAG TO 9)	1700
0617 24 1669 1122	1669	24 ORUMT	ORUMT		CHANGE ORUM TAG TO 9)	1702
1122 65 0178 1077		65 EXITY			MODIFY EXIT)	1704
1077 15 0515 8002		15 110TH	PAKT 8002		MODIFY EXIT)	1706
1241 36 0000 1263		36 0000			WE FOUND A CELL. SHIFT AND COUNT	1708
1263 20 0294 1247	0294	20 XXXX2	XXXX2		STORE THE COUNT	1710
1247 69 0197 1100		69 ALOPT			SHOULD WE RESERVE	1712
1100 90 0755 0905		90 SKP			0, SKIP BECAUSE COMP IS DOUBLE	1714
0905 35 0001 1011		35 0001			RESERVE)	1716
1011 30 0001 0667		30 0001			RESERVE)	1718
0667 11 8003 0975		11 8003			PUT INTO DISTRIBUTOR	1720
0975 35 0004 0685		35 0004			MOVE AMOUNT OF SHIFT INTO 0	1722
0685 10 8001 1291		10 8001			AVAILABILITY WORD BACK INTO UPPER	1724
1291 15 0444 8002		15	8002		MAKE SHIFTING ORDER	1726
0444 30 0000 0717		30 0000			SHIFT MODIFIED WORD BACK INTO PLACE	1728
0717 60 8003 1225		60 8003			CLEAR LOWER	1730
1225 15 0346 0751		15 AI			NEW AVAILABILITY WORD INTO ORUM)	1732
0751 69 0955 1209		69 STR			NEW AVAILABILITY WORD INTO ORUM)	1734
1209 22 0167 8001	0167	22 XXXX1	8001 XXXX1		NEW AVAILABILITY WORD INTO ORUM)	1736
0955 21 9972 0755	9972	21 9972	SKP 9972		NEW AVAILABILITY WORD INTO ORUM)	1738
0755 65 0346 0951		65 AI			ALL CASES. WHAT 010 WE RESERVE)	1740
0951 16 1354 1309		16 600			WHAT CELL 010 WE RESERVE)	1742
1309 30 0004 1220		30 0004			WHAT CELL 010 WE RESERVE)	1744
1220 14 0774 1021		14 41XXX			WHAT CELL 010 WE RESERVE)	1746
1021 16 8002 1079		16 8002			WHAT CELL 010 WE RESERVE)	1748
1079 35 0001 1035		35 0001			WHAT CELL 010 WE RESERVE)	1750
1035 15 8001 1341		15 8001			WHAT CELL 010 WE RESERVE)	1752
1341 10 0294 0649		10 XXXX2			WHAT CELL 010 WE RESERVE)	1754
0649 19 0320 1074		19 901XX			WHAT CELL 010 WE RESERVE)	1756
1074 15 8003 1031		15 8003			WHAT CELL 010 WE RESERVE)	1758
1031 20 0536 0178	0536	20 EQUIV	EXITY EQUIV		STORE THIS ADDRESS IN EQUIV	1760
0915 60 1354 1088		60 A0000	TA			1762
1354 60 1355 1088		60 A0001	TA			1764
0549 60 1356 1088		60 A0002	TA			1766
0599 60 1357 1088		60 A0003	TA			1768
1054 60 1554 1088		60 A0200	TA			1770
0765 60 1555 1088		60 A0201	TA			1772

SUBROUTINE 13. CALCULATE OPTIMUM DYNAMIC LEVEL

					ME0				1774
									1776
0441 24 0179 0532	0179	N SUB13	24 EXITZ		EXITZ	STORE EXIT			1778
0532 95 0735 0787			95	9		D, 0 ADDRESS. I, FORWARD I OR BACKW L			1780
0735 97 0688 0490			97 FD	FIB0		D, FORWARD D. I, BACKWARD D.			1782
0787 97 0490 0742	N 9		97 FIBD	BL		0, FORWARD I. I, BACKWARD L.			1784
0688 65 0095 0699	N FO		65 OPTIM			FORWARD 0. L-D ADDENDS, LEFT ENO LOWE			1786
0699 35 0001 1055			35 0001			L-0 ADOENOS TO LEFT ENO LOWER)			1788
1055 16 8002 1313			16 8002			L-0 ADDENOS TO LEFT END LOWER)			1790
1313 35 0001 1270			35 0001			L-0 ADOENOS TO LEFT END LOWER)			1792
1270 15 8001 1227			15 8001			L-D ADOENOS TO LEFT ENO LOWER)			1794
1227 30 0003 0785			30 0003	SEO		L-0 ADDENDS TO LEFT END LOWER)			1796
0490 65 0095 0749	N FIBD		65 OPTIM			FORWARD I OR BACKWARD 0. GET ADDENOS			1798
0749 92 0402 1105			92 SHOP			0, IT IS A SHIFT OPERATION			1800
1105 93 0660 0710			93 XAS			0, IT IS AN INDEXING OPERATION			1802
0710 35 0002 0785			35 0002	SEO		OTHER. I-ADDOENOS TO LEFT ENO LOWER			1804
0742 65 0197 1051	N BL		65 ALOPT			BACKWARD L. GET ALOPT			1806
1051 92 0402 0656			92 SHOP			D, IT IS A SHIFT OPERATION			1808
0656 93 0660 1061			93 XAS			0, IT IS AN INDEXING OPERATION			1810
1061 91 0364 0688			91	FD		0, IT IS A BRANCH OPERATION			1812
0364 99 0688 0490			99 FO	FIBD		I, BACKWARD L, BRANCH, UNFIXED 0			1814
0402 60 0184 0789	N SHOP		60 P0008			SHIFT OPERATION. FORWARD I, BACKW L.			1816
0789 35 0009 0760			35 0009			ISOLATE LAST DIGIT OF D-ADDRESS)			1818
0760 30 0005 0974			30 0005			ISOLATE LAST DIGIT OF 0-ADDRESS)			1820
0974 10 1277 8003			10	8003		MAKE BRINGING ORDER			1822
1277 65 1250 0706	J		65 1250			BRING SHIFT ADOENOS			1824
0706 69 0095 0548			69 OPTIM			OPTIMIZING TAGS			1826
0548 93 1101 0785			93	SEO		D, OPERATION 31. I, 30,35,36.			1828
1101 35 0004 0785			35 0004	SEO		OPERATION 31. TAKE LARGER ADDENDS.			1830
0660 60 0184 0939	N XAS		60 P0008			INDEXING OP, FORWARD I, BACKWARD L.			1832
0939 30 0004 0799			30 0004			0-ADDRESS TO LEFT ENO OF DISTR)			1834
0799 16 8002 0910			16 8002			0-ADDRESS TO LEFT ENO OF DISTR)			1836
0910 84 1300 1275			84 1300			LOOK UP IN TABLE			1838
1275 15 0428 8002			15	8002		MAKE BRINGING ORDER			1840
0428 65 0000 0756	J		65 0000			BRING ADOENDS			1842
0756 35 0004 0767			35 0004			SHIFT ADOENOS TO LEFT ENO			1844
0767 92 0785 1222			92 SEO			JUMP UNLESS OPERATION 80,82, OR 88			1846
1222 16 0095 0785			16 OPTIM	SEO		REDUCE ADOENOS BY 1 FOR 80,82, OR 88			1848
0785 69 0179 0582	C SEO		69 EXITZ			ALL CASES. WHICH DIRECTION			1850
0582 97 0935 0937			97 P			0, FORWARD. I, BACKWARD.			1852
0937 20 0792 0495	0792		20 IND		IND	BACKWARD. SHOULD WE INTERCHANGE ADOENO			1854
0495 30 0006 0960			30 0006			SHOULD WE INTERCHANGE ADOENOS)			1856
0960 35 0008 0779			35 0008			SHOULD WE INTERCHANGE ADOENOS)			1858
0779 20 0733 0936	0733		20 B		B	SHOULD WE INTERCHANGE ADOENOS)			1860
0936 60 8003 0943			60 8003			SHOULD WE INTERCHANGE ADOENDS)			1862
0943 30 0002 0949			30 0002			SHOULD WE INTERCHANGE ADOENDS)			1864
0949 20 0906 1010	0906		20 A		A	SHOULD WE INTERCHANGE ADOENOS)			1866
1010 15 0733 0987			15 B			SHOULD WE INTERCHANGE ADOENOS)			1868
0987 15 0132 1037			15 12NOX			SHOULD WE INTERCHANGE ADOENOS)			1870
1037 14 0540 1048			14 42NO			SHOULD WE INTERCHANGE ADOENOS)			1872
1048 44 1351 1052			44 OK			I, WE SHOULD INTERCHANGE			1874
1052 66 0906 1111			66 A			INTERCHANGE AND MAKE NEGATIVE)			1876
1111 30 0002 0917			30 0002			INTERCHANGE AND MAKE NEGATIVE)			1878
0917 16 0733 0935			16 B	P		INTERCHANGE AND MAKE NEGATIVE)			1880
1351 66 0792 0935	N OK		66 IND	P		MAKE NEGATIVE, BUT DO NOT INTERCHANGE			1882
0935 20 0167 1320	0167	C P	20 XXXX1		XXXX1	ALL CASES. STORE ADOENOS			1884
1320 65 0062 0967			65 OPREG			WHICH ONE DO WE USE)			1886
0967 14 1022 1060			14 21XXX			WHICH ONE DO WE USE)			1888
1060 44 1014 0414			44	EVN		0, OPREG WAS OOD. I, EVEN.			1890
1014 65 0167 0771			65 XXXX1			ODD. SEPARATE LAST ADOENO)			1892
0771 35 0002 1327			35 0002			OOD. SEPARATE LAST ADOEND.			1894
1327 65 8002 0985			65 8002	BOTH		OOD. SEPARATE LAST ADOENO.			1896
0414 65 0167 0985	N EVN		65 XXXX1	BOTH		EVEN. WE SHOULD USE FIRST ADOEND			1898
0985 30 0008 0956	C BOTH		30 0008			BOTH. RIGHT ADOENO TO RIGHT ENO			1900
0956 15 0062 1017			15 OPREG			ADD OLO DYNAMIC LEVEL			1902
1017 15 0320 1325			15 501XX			ADD 50 TO MAKE SURELY POSITIVE			1904
1325 69 0179 0632			69 EXITZ			IF BACKWARD L, SUBTRACT TAG-COUNT)			1906
0632 95 1085 1087			95 CP			IF BACKWARD L, SUBTRACT TAG-COUNT)			1908
1087 97 1085 0942			97 CP			IF BACKWARD L, SUBTRACT TAG-COUNT)			1910
0942 16 0545 1085			16 COUNT	CP		IF BACKWARD L, SUBTRACT TAG-COUNT)			1912
1085 14 0320 1272	C CP		14 501XX			NEW LEVEL, MODULO 50 IN UPPER			1914
1272 65 8003 0179			65 8003	EXITZ		NEW DYNAMIC LEVEL INTO LOWER			1916
0540 04 0000 0000	K 42ND		04 0000	0000					1918

SUBROUTINE 14. CONTROL INFO, OP, OPTIM, ALOPT.

HED

1920
1922

0039	24	0177	0430	0177	N SUB14	24	EXITX		EXITX	STORE EXIT	1924
0430	69	0154	1110			69	R0004			CONTROL INFORMATION FOR PUNCHING)	1926
1110	24	0186	0989	0186		24	P0010		P0010	CONTROL INFORMATION FOR PUNCHING)	1928
0989	65	0155	1210			65	R0005			BRING TAGS AND OPERATION	1930
1210	69	8003	0566			69	8003			CLEAR DISTRIBUTOR	1932
0566	22	0183	0986	0183		22	P0007		P0007	STORE OPERATION FOR PUNCHING	1934
0986	16	8001	0993			16	8001			CLEAR OP FROM ACCUMULATOR	1936
0993	35	0005	1006			35	0005			D TAG INTO UPPER	1938
1006	21	0902	1056	0902		21	0902		0902	STORE D TAG	1940
1056	65	8002	0965			65	8002			DELETE D TAG	1942
0965	30	0005	0628			30	0005			I TAG TO RIGHT END LOWER	1944
0628	20	0903	1106	0903		20	0903		0903	STORE I TAG	1946
1106	45	1260	1211			45		BT		D, I TAG IS NON-ZERO	1948
1260	65	0515	1211			65	110TH	BT		COUNT OF I TAGS IN LOWER	1950
1211	10	0902	1310		C BT	10	0902			D TAG INTO UPPER	1952
1310	44	0464	0514			44		UM		O, D TAG IS NON-ZERO	1954
0464	15	0515	0514			15	110TH	UM		COUNT OF TAGS IN LOWER	1956
0514	20	0545	0598	0545	C UM	20	COUNT		COUNT	STORE TAG-COUNT	1958
0598	60	0183	1237			60	P0007			OPERATION IN D OF UPPER	1960
1237	10	0590	8003			10		8003		MAKE BRINGING ORDER	1962
0590	65	0800	1306		J	65	0800			BRING OPTIMIZING ADDENDS AND TAGS	1964
1306	24	0095	0648	0095		24	OPTIM		OPTIM	STORE OPTIMIZING ADDENDS AND TAGS	1966
0648	69	8003	1261			69	8003			CLEAR DISTRIBUTOR	1968
1261	23	0167	0971	0167		23	XXXX1		XXXX1	STORE LAST 4 DIGITS OF OPTIM	1970
0971	35	0001	0678			35	0001			ONE DIGIT INTO UPPER	1972
0678	44	1081	0682			44	YES			I, O IS NOT TRUE ADDRESS	1974
0682	60	1235	1089			60	8	AX		MORE TAGS INTO UPPER	1976
1081	60	0584	1089		N YES	60	9	AX		MORE TAGS INTO UPPER	1978
1089	10	0167	1071		C AX	10	XXXX1			AOD IN TAGS FROM OPTIM	1980
1071	21	0197	0177	0197		21	ALOPT	EXITX	ALOPT	STORE ALL OPTIMIZING TAGS	1982
1235	98	8008	0000		K 8	98	8008	0000			1984
0584	98	8009	0000		K 9	98	8009	0000			1986

SUBROUTINE 17. MODIFY DYN LEV IF D = 8001+8002+8003

HED

1988
1990

0586	24	0178	1231	0178	N SUB17	24	EXITY		EXITY	STORE EXIT INSTRUCTION	1992
1231	20	0294	1297	0294		20	XXXX2		XXXX2	STORE DYNAMIC LEVEL, RIGHT END WORD	1994
1297	14	1022	1322			14	21XXX			DIVIDE DYNAMIC LEVEL BY 2	1996
1322	44	1076	1026			44		EVN		D, DYNAMIC LEVEL IS ODD.	1998
1076	65	1279	1033			65	8002	E0		ODD. PUT 8002 AT RIGHT END OF LOWER	2000
1026	65	1229	1033		N EVN	65	8003	E0		EVEN. PUT 8003 AT RIGHT END OF LOWER	2002
1033	16	0536	0992		C E0	16	EQUIV			BOTH. SUBTRACT THE 800X ADDRESS	2004
0992	24	0184	1287	0184		24	P0008		P0008	STORE THE ADDRESS FOR PUNCHING	2006
1287	45	0640	1042			45	XY			I, 8002-8003 + WRONG PARITY	2008
1042	65	0515	0178			65	110TH	EXITY		UNIT CORRECTION TO DYNAMIC LEVEL	2010
0640	65	1043	1347		N XY	65	8001			8001 INTO RIGHT END LOWER	2012
1347	16	0536	1092			16	EQUIV			SUBTRACT THE ADDRESS IN QUESTION	2014
1092	45	0546	0698			45	NZ			D, ADDRESS IS NOT 8001	2016
0698	66	0515	0178			66	110TH	EXITY		IF 8001, CORRECTION IS -1	2018
0546	67	8003	0178		N NZ	67	8003	EXITY		IF NOT 8001, CORRECTION IS ZERO	2020
1043	00	0000	8001		K 8001	00	0000	8001			2022
1279	00	0000	8002		K 8002	00	0000	8002			2024
1229	00	0000	8003		K 8003	00	0000	8003			2026

SUBROUTINE 18. INOEX FOR USE WITH INDEXING REGISTER

HED

2028
2030

0677	24	0178	1281	0178	N INDEX	24	EXITY		EXITY	STORE EXIT	2032
1281	20	0167	1121	0167		20	XXXX1		XXXX1	STORE ADDRESS	2034
1121	65	0178	0783			65	EXITY			D-POSITION HOLDS 0 FOR O, 1 FOR I)	2036
0783	30	0004	1093			30	0004			D-POSITION HOLOS 0 FOR D, 1 FOR I)	2038
1093	35	0004	1311			35	0004			D-POSITION HOLOS 0 FOR D, 1 FOR I)	2040
1311	15	0564	8002			15		8002		BRINGING ORDER	2042
0564	65	0902	0662		J	65	0902			BRING APPROPRIATE TAG	2044
0662	15	8002	1221			15	8002			DOUBLE IT	2046
1221	10	0167	1271			10	XXXX1			INOEXABLE ADDRESS INTO UPPER	2048
1271	11	1124	0929			11	27TH			SUBTRACT 2000, TO DETERMINE RANGE	2050
0929	46	0732	0933			46	DRM			I, CORE. D, DRUM.	2052
0933	11	8003	1242			11	8003			CORE. CLEAR UPPER	2054
1242	30	0001	0999			30	0001	A		SHIFT RIGHT FOR CORE ADDENO	2056

0732 10 8001 0999	N DRM	10 8001	A	DRUM. MAKE POSITIVE AGAIN	2058
0999 15 0167 0178	C A	15 XXXX1	EXITX	ADD ADDRESS BEING INDEXED. EXIT.	2060
1124 00 0000 2000	K 27TH	00 0000	2000	FOR TELLING WHETHER DRUM OR CORE	2062

SUBROUTINE 19. PROCESS L FORWARD

HED					2064
					2066
0042 24 0177 0480	0177 N SUB19	24 EXITX		EXITX STORE EXIT	2068
0480 60 0151 0712		60 R0001		ALPHABETICAL L TO UPPER	2070
0712 69 1015 0063		69 PRE	SUBR9	ANALYZE L-ADDRESS	2072
1015 65 0536 0421	J PRE	65 EQUIV	L0004	BRING EQUIVALENT. MULTIBRANCH	2074
0418 69 1015 0661	N L0001	69 PRE	SUBR7	SUB 11 FOUND EQU OF SYMB. RESERVE	2076
0419 60 0622 0627	R L0002	60 87THX	SB10B	N.G. FIX TO OMIT PUNCHING	2078
0420 69 1669 1224	R L0003	69 DRUMT		BLANK ADDRESS. IS DRUM FULL	2080
1224 90 0728 0419		90	L0002	D, DRUM NOT FULL. I, DRUM IS FULL.	2082
0728 65 1023 0421		65 BLANK	L0004	USE PROPER VALUE FOR BLANK ADDRESS	2084
0421 20 0182 1285	0182 N L0004	20 P0006		DRUM. STORE ADDRESS FOR PUNCHING	2086
1285 60 8001 1292		60 8001		PUT IT INTO UPPER	2088
1292 69 0595 1025		69	SUB2R	RESERVE ADDRESS	2090
0595 65 0182 1337		65 P0006	MW	BRING BACK NUMERICAL ADDRESS	2092
0422 69 1030 0983	R L0005	69 1030		NEW SYMBOL. WHERE ARE WE	2094
0983 90 1001 0738		90 MDPLL		D, START SEARCH. I, WE ARE ON OUITT	2096
0738 65 1342 0611		65	SUB11	FIND AN EQUIVALENT	2098
1342 00 0890 0418	J	00 0890	L0001	TAGS AND EXIT FOR USE IN SUBR 11	2100
0423 00 0000 0001	R L0006	00 0000	SETCC	800X ADDRESS. OUIT IMMEOIATELY	2102
0424 20 0182 1335	0182 R L0007	20 P0006		P0006 CORE ADDRESS. STORE FOR PUNCHING	2104
1335 65 0292 1337		65 ORCEO	MW	DYNAMIC LEVEL OF LAST CORE ADDRESS	2106
1337 15 0545 0335	C MW	15 COUNT	SB10C	CORE OR DRUM. AOD TAG-COUNT TO DYN LV	2108
0425 00 0000 0419	R L0008	00 0000	L0002	MISCELLANEOUS ADDRESS. OUIT AND OMIT	2110

SUBROUTINE 20. PROCESS D FORWARD

HED					2112
					2114
0045 24 0177 0630	0177 N SUB20	24 EXITX		EXITX STORE EXIT	2116
0630 60 0152 0762		60 R0002		ALPHABETICAL D INTO UPPER	2118
0762 69 1065 0063		69 PRE	SUBR9	ANALYZE D-ADDRESS	2120
1065 65 0536 0472	J PRE	65 EQUIV	D0004	BRING EQUIVALENT. MULTIBRANCH	2122
0469 20 1023 0476	1023 N D0001	20 BLANK	D0008	BLANK STORE EQUIV OF BLANK FOR REFERENCE	2124
0470 60 0672 0627	R D0002	60 88THX	SB10B	N.G. OUIT AND OMIT PUNCHING	2126
0471 69 0095 0748	R D0003	69 OPTIM		BLANK D. IS IT MDF OPERATION	2128
0748 94 0800 0912		94 0800		IF BLANK D AND MDF, STOP MACHINE	2130
0912 65 1115 0611		65	SUB11	BLANK D, BUT NOT MDF. FIND VALUE OF D	2132
1115 00 0888 0469	J	00 0888	D0001	TAGS AND EXIT FOR SUBROUTINE 11	2134
0472 69 1226 0677	N D0004	69	INDEX	DRUM ADDRESS. INDEX IF TAGGED	2136
1226 00 0000 0476	J	00 0000	D0008	ADDEND AND EXIT FOR USE IN SUBR 18	2138
0473 65 1276 0611	R D0005	65	SUB11	NEW SYMBOL. FIND EQUIVALENT	2140
1276 00 0888 0477	J	00 0888	D0009	TAGS AND EXIT FOR USE IN SUBR 11	2142
0474 69 0778 0441	R D0006	69	SUB13	800X ADDRESS. FIND DYNAMIC LEVEL	2144
0778 00 0808 0782	J	00 0808		TAGS AND EXIT FOR USE IN SUBR 11	2146
0782 69 1086 0586		69	SUB17	PERHAPS MODIFY DYNAMIC LEVEL)	2148
1086 15 0294 1049	J	15 XXXX2	B	PERHAPS MODIFY DYNAMIC LEVEL)	2150
0475 69 0928 0677	R D0007	69	INDEX	CORE ADDRESS. INDEX IF TAGGED	2152
0928 00 0000 0932	J	00 0000		AODEND AND EXIT FOR SUBROUTINE 18	2154
0932 20 0184 0788	0184	20 P0008		P0008 STORE THE CORE ADDRESS FOR PUNCHING	2156
0788 69 1243 0441		69	SUB13	FIND DYNAMIC LEVEL	2158
1243 00 0808 0798	J	00 0808		TAGS AND EXIT FOR SUBROUTINE 13	2160
0798 20 0292 1049	0292	20 ORCEO	B	ORCEO STORE DYNAMIC LEVEL FOR REFERENCE	2162
0476 20 0184 1049	0184 R D0008	20 P0008	B	P0008 MISC ADDRESS, USUALLY A CONSTANT	2164

1049 69 0095 0948	C B	69 OPTIM		SEVERAL CASES. WHAT KIND OF OP	2166
0948 91 0177 0335		91 EXITX	SB10C	O, BRANCH. I, ARITHMETIC.	2168
0477 69 1065 0661	N 00009	69 PRE	SUBR7	STORE EQUIVALENT OF NEW SYMBOL	2170
0478 00 0000 0470	R 00010	00 0000	00002	SYMB TABLE FULL OR DRUM PACKED. QUIT	2172

SUBROUTINE 21. PROCESS I FORWARD

HEO

0018 24 0177 0680	0177	N SUB21	24 EXITX		EXITX STORE EXIT	2174
0680 60 0153 0962			60 R0003		ALPHABETIC I TO UPPER	2176
0962 69 1215 0063			69	SUBR9	ANALYZE I-ADDRESS	2178
1215 65 0536 0575		J	65 EQUIV	I0004	BRING EQUIVALENT. MULTIBRANCH	2180
0572 20 1023 0577	1023	N I0001	20 BLANK	I0006	BLANK STORE EQUIVALENT OF BLANK FOR REFERENCE	2182
0573 60 0722 0627		R I0002	60 89THX	SB10B	N.G. QUIT AND OMIT PUNCHING	2184
0574 60 0152 1012		R I0003	60 R0002		BLANK I. IS D ALSO BLANK	2186
1012 44 1265 0616			44 DNB		O, O IS NOT BLANK.	2188
0616 69 1669 1274			69 DRUMT		BLANK O AND I. CHECK DRUM TAG	2190
1274 90 0978 0573			90	I0002	I, DRUM IS FULL	2192
0978 65 1023 0579			65 BLANK	I0008	MAKE BLANK I EQUAL TO BLANK O	2194
1265 65 0768 0611		N DNB	65	SUB11	USUAL CASE, I BLANK AND NOT O.	2196
0768 00 0889 0572		J	00 0889	I0001	TAGS AND EXIT FOR SUBROUTINE 11	2198
0575 69 1028 0677		N I0004	69	INOEX	DRUM ADDRESS. INOEX IF TAGGED	2200
1028 00 0001 0577		J	00 0001	I0006	ADDRESS AND EXIT FOR USE IN SUBR 18.	2202
0576 65 0979 0611		R I0005	65	SUB11	NEW SYMBOL. FIND BEST VALUE.	2204
0979 00 0889 0580		J	00 0889	I0009	TAGS AND EXIT FOR SUBR 11.	2206
0577 20 0185 0177	0185	R I0006	20 P0009	EXITX	P0009 800X ADDRESS. STORE AND EXIT	2208
0578 69 1331 0677		R I0007	69	INOEX	CORE ADDRESS. INOEX IF TAGGED	2210
1331 00 0001 1236		J	00 0001		ADDRESS AND EXIT FOR SUBROUTINE 18.	2212
1236 20 0185 0938	0185		20 P0009		P0009 STORE CORE ADDRESS FOR PUNCHING	2214
0938 69 1293 0441			69	SUB13	FIND DYNAMIC LEVEL	2216
1293 00 0809 0998		J	00 0809		TAGS AND EXIT FOR SUBROUTINE 13	2218
0998 20 0292 0177	0292		20 ORCEO	EXITX	ORCEO STORE DYNAMIC LEVEL FOR REFERENCE	2220
0579 20 0185 0177	0185	R I0008	20 P0009	EXITX	P0009 OTHER ADDRESS. STORE AND EXIT.	2222
0580 20 0185 0988	0185	N I0009	20 P0009		P0009 STORE EQUIVALENT OF NEW SYMBOL	2224
0988 69 0177 0661			69 EXITX	SUBR7	STORE SYMBOL AND ITS EQUIVALENT	2226
0581 00 0000 0573		R I0010	00 0000	I0002	SYMBOL TABLE FULL. OMIT PUNCHING ADDRESS	2228

SUBROUTINE 22. SAVE ORCEO ON FIRST BACKWARD CARD

0201 69 1050 1062	N SUB22	69 1050		FIRST-CARD TAG	2234
1062 90 8002 1067		90 8002		O, WE ARE NOT ON FIRST BACKWARD CARD	2236
1067 69 0292 0645		69 ORCEO		STORE ORCEO FOR USE WHEN WE START FORWARD	2238
0645 24 0486 8002	0486	24 SAVOR	8002	SAVOR STORE ORCEO FOR USE WHEN WE START FORWARD	2240

ROUTINE FOR TYPE 2 CARD, NAMELY RELOCATABLE SUBROUTINE

HED

0002 69 1112 0039	R	0002	69	SUB14	OP, C, I, TAGS, OPTIM, ALOPT	2244
1112 65 0157 1212	J		65 R0007		NUMERICAL L	2246
1212 10 0151 1262			10 R0001		ALPHABETIC L	2248
1262 69 1315 1024			69	SUBR8	PROCESS L	2250
1315 44 1321 1324	J		44 NGL		O, RELOCATED L IS EXCESSIVE	2252
1324 20 0182 1286	0182		20 P0006	PROD	P0006 STORE L FOR PUNCHING	2254
1321 10 0186 1343	N NGL		10 P0010		FIX C, I. SO THAT L WILL NOT PUNCH	2256
1343 21 0186 1286	0186		21 P0010	PROD	FIX C, I. SO THAT L WILL NOT PUNCH	2258
1286 65 0158 0614	C PROD		65 R0008		NUMERICAL D	2260
0614 10 0152 1312			10 R0002		ALPHABETIC, O	2262
1312 69 0666 1024			69	SUBR8	PROCESS D	2264
0666 44 1326 1078	J		44 NGD		D, RELOCATED D IS EXCESSIVE	2266

1078 69 0982 0677			69	INDEX	INDEX D	2272
0982 00 0000 1336	0184	J	00 0000		ADDEND AND EXIT FOR INDEXING	2274
1336 20 0184 1238			20 P0008	PRDI	STORE D FOR PUNCHING	2276
1326 30 0001 1083		N NGD	30 0001		FIX C.I. SD THAT D WILL NOT PUNCH)	2278
1083 10 0186 0494	0186		10 P0010		FIX C.I. SD THAT D WILL NOT PUNCH)	2280
0494 21 0186 1238			21 P0010	PRDI	FIX C.I. SD THAT D WILL NOT PUNCH)	2282
1238 65 0159 0664		C PRQI	65 R0009		NUMERICAL I	2284
0664 10 0153 0714			10 R0003		ALPHABETIC I	2286
0714 69 1117 1024			69	SUBR8	PRDCESS I	2288
1117 44 1228 1278		J	44 NGI		D,RELOCATED I IS EXCESSIVE	2290
1278 69 1032 0677			69	INDEX	INDEX I	2292
1032 00 0001 1288		J	00 0001		ADDEND AND EXIT FOR INDEXING	2294
1288 20 0185 1338	0185		20 P0009	ALL	STORE I FOR PUNCHING	2296
1228 30 0002 1239		N NGI	30 0002		FIX C.I. SD THAT I WILL NOT PUNCH)	2298
1239 10 0186 0544	0186		10 P0010		FIX C.I. SD THAT I WILL NOT PUNCH)	2300
0544 21 0186 1338			21 P0010	ALL	FIX C.I. SD THAT I WILL NOT PUNCH)	2302
1338 60 0003 0057		C ALL	60 READC	SB10A	PREPARE TO PUNCH AND READ NEXT CARD	2304

EQU, SYN, AND RED ROUTINES

			HED				2306
			EDU E				2308
				DPTIM	DDUBLE-PURPDSE TO CONSERVE DRUM SPACE		2310
0054 60 0153 0764	R	0054	60 R0003	BDTH	EDU ENTRY. ALPHABETIC I		2312
0025 60 0153 0764	R	0025	60 R0003	BDTH	SYN ENTRY. ALPHABETIC I		2314
0764 69 1217 0063	C	BDTH	69	SUBR9	ANALYZE I		2316
1217 65 0536 0120	J		65 EDUIV	D0003	BRING EDUIVALENT. MULTIBRANCH.		2318
0118 65 1328 1233	R	D0001	65 83RD	BP	INVALID ADDRESS		2320
0119 65 1328 1233	R	D0002	65 83RD	BP	BLANK I. CARD IS UNUSABLE		2322
0120 20 0095 1098	0095	N D0003	20 E	X	E DRUM. PUT I-EDUIVALENT INTO E		2324
0121 65 1328 1233	R	D0004	65 83RD	BP	NEW SYMBDL I. CARD IS UNUSABLE		2326
0122 20 0095 1098	0095	R D0005	20 E	X	E 800X ADDRESS		2328
0123 20 0095 1098	0095	R D0006	20 E	X	E CORE ADDRESS		2330
0124 20 0095 1098	0095	R Q0007	20 E	X	E OTHER ADDRESS		2332
1098 60 0152 0914	C	X	60 R0002		ALPHABETIC D		2334
0914 69 1267 0063			69	SUBR9	ANALYZE D		2336
1267 65 0095 0127	J		65 E	D0010	BRING VALUE OF I INTO LOWER		2338
0125 65 1328 1233	R	Q0008	65 83RD	BP	N G D. CARD IS UNUSABLE		2340
0126 65 1328 1233	R	D0009	65 83RD	BP	BLANK D. CARD IS UNUSABLE		2342
0127 20 0536 1289	0536	N D0010	20 EQUIV	SD	EDUIV DLO SYMBDL. STORE I AS NEW EDUIV		2344
0128 20 0536 1289	0536	R D0011	20 EDUIV	SD	EDUIV NEW SYMBQL. STORE I AS ITS EDUIVALENT		2346
0129 20 0536 1289	0536	R D0012	20 EQUIV	SD	EDUIV 800X. STORE ITS EDUIVALENT.		2348
0130 20 0536 1289	0536	R D0013	20 EQUIV	SD	EDUIV CORE. STORE I AS ITS EDUIVALENT.		2350
0131 20 0536 1289	0536	R D0014	20 EQUIV	SD	EDUIV DTHER. STORE I AS THE EQUIVALENT.		2352
1289 69 0594 0661		C SD	69	SUBR7	STORE SYMBDL AND EDUIVALENT IN TABLE		2354
0594 60 0095 1099	J		60 E		BRING SYMBDL-EQUIVALENT TO UPPER		2356
1099 69 0001 1025			69 SETCC	SUB2R	RESERVE AND PUNCH CARD		2358
1233 10 0003 0709	N	BP	10 REAQC	SUB10	BY-PASS INVALID CARD		2360
0098 65 0159 0964	R	0098	65 R0009		RED ENTRY. NUMERICAL I		2362
0964 16 0409 1064			16 2000I		CORE DR DRUM		2364
1064 46 1317 0968			46 D		D, DRUM. I, CORE.		2366
0968 15 0726 1082			15 CDIFF	CD	ADD CORE RELOCATION AMOUNT		2368
1317 15 1019 1082	N	D	15 DDIFF	CD	ADD DRUM RELOCATION AMOUNT		2370
1082 15 0409 0124	C	CD	15 2000I	D0007	RESTORE THE 2000 AND JUMP BACK		2372
1328 00 8000 0000	K	83RD	00 8000	0000			2374

RBD ROUTINE

			HED				2376
0094 60 0158 1114	R	0094	60 R0008		BEGINNING OF REGION TO BE RESERVED		2380

1114 69 1018 1025		69	SUB2R	RESERVE FIRST CELL IN BANO	2382
1018 60 0153 1214	J	60 R0003		IS THE I-ADDRESS BLANK	2384
1214 45 1068 1329		45	OUT	0, NOT BLANK. I, IT IS BLANK.	2386
1068 60 0158 1264		60 R0008		BRING FIRST ADDRESS AGAIN	2388
1264 10 0320 0730		10 501XX		ADVANCE IT	2390
0730 21 0198 1314	0158	21 R0008		STORE ADVANCEO O ADDRESS	2392
1314 11 0159 0716		11 R0009		SUBTRACT ENO OF AREA TO BE RESERVED	2394
0716 61 8003 0780		61 8003		CHANGE ITS SIGN	2396
0780 46 1329 0094		46 OUT	0094	0, WE HAVE FINISHED	2398
1329 70 0151 1102	N OUT	70 R0001		WHEN ONE, READ NEXT CARD	2400
1102 65 0160 0766		65 R0010		IS THIS ANOTHER RBO CARD	2402
0766 16 0930 1339		16 94I		IS THIS ANOTHER RBO CARD	2404
1339 45 0644 0694		45 ALL		I, IT IS ANOTHER RBO CARD	2406
0694 60 8001 0709		60 8001	SUB10	PUNCH A CARD, AND REPEAT RBO	2408
1034 16 0038 0744	A	16 16THX		DUPLICATE INTO AVAILABILITY TABLE	2410
0744 11 8001 0452		11 8001		DUPLICATE INTO AVAILABILITY TABLE	2412
0452 10 0916 0980		10 Q		DUPLICATE INTO AVAILABILITY TABLE	2414
0980 46 1283 0634		46 00NE		DUPLICATE INTO AVAILABILITY TABLE	2416
0634 11 8001 8002		11 8001	8002	DUPLICATE INTO AVAILABILITY TABLE	2418
0644 61 1248 0966	N ALL	61 SENO		DUPLICATE INTO AVAILABILITY TABLE	2420
0966 16 1230 8002		16	8002	DUPLICATE INTO AVAILABILITY TABLE	2422
1230 69 1355 8003	J	69 A0001	8003	DUPLICATE INTO AVAILABILITY TABLE	2424
1248 21 1359 1034	1359 J SENO	24 A0005	A A0005	DUPLICATE INTO AVAILABILITY TABLE	2426
1283 65 1680 0690	N 00NE	65 ZEROX		CLEAR ACCUMULATOR	2428
0690 69 0794 1025		69	SUB2R	RESERVE 0000	2430
0794 60 0160 0709		60 R0010	SUB10	PUNCH CARD AND PROCEED	2432
0916 24 1555 0000	1555 Q Q	24 A0201	0000 A0201		2434
0930 00 0000 0094	K 94I	00 0000	0094		2436

BLR, BLA, AND RBR ROUTINES

HEO

0029 69 1232 0740	R	0029 69 ZQ	SUB	BLR ENTRY. TO SUBROUTINE	2442
1232 69 0001 1025	J ZQ	69 SETCC	SUB2R	TO RESERVING SUBROUTINE	2444
0021 69 1044 1298	R	0021 69 81STX		BLA ENTRY. CHANGE ORUM TAG	2446
1298 24 1669 1280	1669	24 ORUMT	ORUMT	CHANGE ORUM TAG. ORUM CANNOT BE FULL.	2448
1280 69 1333 0740		69	SUB	TO SUBROUTINE	2450
1333 69 0001 1075	J	69 SETCC	SUB2U	TO UNRESERVING SUBROUTINE	2452
0099 69 0502 0740	R	0099 69	SUB	RBR ENTRY. TO SUBROUTINE	2454
0502 10 1019 1232	J	10 00IFF	ZQ	ADD RELOCATION AMOUNT	2456
0740 24 0179 1282	0179 N SUB	24 EXITZ	EXITZ	STORE EXIT	2458
1282 65 0153 1016		65 R0003		ALPHABETICAL I	2460
1016 45 1330 1332		45	BL	I, I IS UNPUNCHED	2462
1330 65 0159 1066		65 R0009		NUMERICAL I INTO LOWER	2464
1066 16 0158 1332		16 R0008	BL	SUBTRACT NUMERICAL 0 FROM LOWER	2466
1332 10 0158 0179	C BL	10 R0008	EXITZ	NUMERICAL 0 INTO UPPER	2468

REG ROUTINE

HEO

0097 65 0152 1116	R	0097 65 R0002		REGION DESIGNATOR TO RIGHT ENO	2474
1116 30 0008 0790		30 0008		REGION DESIGNATOR TO RIGHT ENO	2476
0790 16 1308 1216		16 99THX		SUBTRACT 90	2478
1216 46 0684 0029		46	0029	IF A DIGIT, TO BLR WITHOUT DESIGNATING	2480
0684 15 1119 0734		15 29IXX		ADD 29	2482
0734 46 0029 0940		46 0029		IF SPEC CHAR, TO BLR DIRECTLY	2484
0940 35 0004 0552		35 0004		SHIFT TO Q POSITION	2486
0552 15 1266 0784		15 ST		MAKE STORING ORDER	2488
0784 69 0158 8002		69 R0008	8002	ADDRESS OF 0001 OF REGION	2490
1266 21 1660 0029	1660 J ST	24 G0001	0029 G0001	STORE ADDRESS OF 0001 OF REGION	2492

HEO ROUTINE

2494

HEO										2496
0084 65 0990 8002	R	0084 65	8002	ADDDING ORDER TO LOWER	2498					
0990 10 1898 0608	J	10 ZMAXM	U0001	ADD A SYMBOL TO UPPER	2500					
0608 44 1316 0001	R	U0001 44	SETCC	I, IT IS ZERO AND WE HAVE FINISHED	2502					
1316 35 0008 1040		35 0008		CLEAR ALL EXCEPT LAST CHARACTER)	2504					
1040 30 0008 1118		30 0008		CLEAR ALL EXCEPT LAST CHARACTER)	2506					
1118 44 0001 0934		44 SETCC		O, QUIT BECAUSE WE HAVE FOUND LONG SYM	2508					
0934 15 1090 8002		15 K	8002	MAKE STORING ORDER, AND STORE ZERO	2510					
0609 16 1218 8002	R	U0002 16 KA	8002	RESTORE LOWER, AND ADD NEXT SYMBOL	2512					
1090 11 0000 0001	P	K 11 0000	0001	TO CHANGE ADDING ORDER TO STORING ONE	2514					
1218 11 0001 0001	P	KA 11 0001	0001	TO CHANGE STORING ORDER TO ADDING ONE	2516					

ALF ROUTINE										2518
HEO										2520
0016 65 0114 0984	R	0016 65 91STX		TAG IN CASE L IS UNDEFINED SYMBOL)	2522					
0984 24 1030 1084 1030		24 1030		1030 TAG IN CASE L IS UNDEFINED SYMBOL)	2524					
1084 24 0197 1350 0197		24 ALOPT		ALOFT TAG IN CASE L IS UNDEFINED SYMBOL)	2526					
1350 69 0154 1268		69 R0004		CONTROL INFORMATION)	2528					
1268 24 0186 1240 0186		24 P0010	P0010	CONTROL INFORMATION)	2530					
1240 69 0944 0042		69	SUB19	PROCESS L, IF REGIONAL OR SYMBOLIC	2532					
0944 65 0152 1318	J	65 R0002		ALPHABETIC SYMBOL	2534					
1318 24 0185 1290 0185		24 P0009	P0009	STORE LAST 4 DIGITS OF CONSTANT IN I	2536					
1290 30 0004 0602		30 0004			2538					
0602 20 0184 1340 0184		20 P0008	P0008	STORE NEXT 4 DIGITS IN O POSITION	2540					
1340 24 0183 0994 0183		24 P0007	P0007	STORE FIRST 2 DIGITS IN OP POSITION	2542					
0994 60 0003 0057		60 REAOC	SB10A	PUNCH	2544					

PAT ROUTINE										2546
HEO										2548
0073 60 0010 0709	R	0073 60 0010	SUB10	PUNCH USUAL DUMMY OUTPUT CARO	2550					
0010 69 1234 1244	J	0010 69 85TH		C.I. FOR AVAILABILITY TABLE)	2552					
1244 24 0186 1294 0186		24 P0010	P0010	C.I. FOR AVAILABILITY TABLE)	2554					
1294 60 1348 1284		60 A1		VARIABLE BRINGING ORDER INTO UPPER	2556					
1284 15 1344 1249		15 RS1	LOOP	INITIAL WORD SHOWING LOCATION OF TABLE	2558					
1249 20 0177 1334 0177	C	LOOP 20 P0001	P0001	STORE LOCATION OF AVAILABILITY WORD	2560					
1334 15 0695 1299		15 C3		MODIFY IDENTIFICATION OF WORD	2562					
1299 20 0179 0745 0179		20 P0003	P0003	STORE LOCATION OF AVAILABILITY WORD	2564					
0745 15 0695 1349		15 C3		MODIFY IDENTIFICATION OF WORD	2566					
1349 20 0185 0795 0185		20 P0009	P0009	STORE LOCATION OF AVAILABILITY WORD	2568					
0795 15 0695 0652		15 C3		MODIFY IDENTIFICATION OF WORD	2570					
0652 20 0183 8003 0183		20 P0007	8003	P0007 STORE LOCATION OF AVAILABILITY WORD	2572					
0183 24 0178 0945 0178	R	T0001 24 P0002	P0002	STORE AVAILABILITY WORD	2574					
0945 10 0702 8003		10 C6	8003	MODIFY BRINGING ORDER, AND BRING	2576					
0169 24 0180 0995 0180	R	T0002 24 P0004	P0004	STORE AVAILABILITY WORD	2578					
0995 10 0702 8003		10 C6	8003	MODIFY BRINGING ORDER, AND BRING	2580					
0170 24 0182 1045 0182	R	T0003 24 P0006	P0006	STORE AVAILABILITY WORD	2582					
1045 10 0702 8003		10 C6	8003	MODIFY BRINGING ORDER, AND BRING	2584					
0171 24 0184 1095 0184	R	T0004 24 P0008	P0008	STORE AVAILABILITY WORD	2586					
1095 10 0702 8003		10 C6	8003	MODIFY BRINGING ORDER, AND BRING	2588					
0172 71 0177 1096	R	T0005 71 P0001		PUNCH A CARO OF THE TABLE	2590					
1096 11 1352 1046		11 O		ARE WE DONE	2592					
1046 44 0752 0003		44	REAOC	O, WE ARE NOT DONE.	2594					
0752 10 1245 0952		10 OS		RESTORE VARIABLE BRINGING ORDER	2596					
0952 16 1295 1249		16 C7	LOOP	MODIFY IDENTIFICATION FOR NEXT CARO	2598					
1348 69 1355 0168	P	A1 69 A0001	T0001	INITIAL VARIABLE BRINGING ORDER	2600					
0702 00 0001 0001	P	C6 00 0001	0001	TO MODIFY VARIABLE BRINGING ORDER	2602					
1352 69 1555 0172	O	O 69 A0201	T0005	COMPARISON CONSTANT FOR ENO OF JOB	2604					
1245 69 1555 0168	O	OS 69 A0201	T0001	TO RESTORE BEFORE ENO OF JOB	2606					
1344 00 0000 0450	K	RS1 00 0000	0450	INITIAL IDENTIFICATION OF WORD	2608					
0695 00 0500 0500	K	C3 00 0500	0500	TO MODIFY IDENTIFICATION	2610					
1295 00 1499 1499	K	C7 00 1499	1499	TO MODIFY IDENTIFICATION AT ENO OF LI	2612					
1234 00 0080 0000	K	85TH 00 0080	0000		2614					

										REL ROUTINE	
										HED	
0093	65	0152	1345		R	0093	65	R0002		ALPHABETIC D	2616
1345	45	0596	0646				45		S00	IF 0 IS BLANK, WE SHOULD STORE ZERO	2618
0596	65	0158	0646				65	R0008	SD0		2620
0646	20	1019	0696	1019	C	S00	20	00IFF		00IFF	2622
0696	65	0153	0746				65	R0003		BRING NUMERICAL 0	2624
0746	45	0796	0946				45		SC0	ALPHABETIC I	2626
										JUMP IF I IS BLANK	2628
0796	65	0159	0946				65	R0009	SC0	BRING NUMERICAL I	2630
0946	20	0726	0001	0726	C	SC0	20	00IFF	SETCC	00IFF	2632
										STORE CORE RELOCATION AMOUNT	2634

										BOP ROUTINE	
0027	69	0001	0006		R	0027	69	SETCC	SUBR1	TO INITIALIZING SUBROUTINE	2636
											2638

										GENERAL CONSTANTS	
0515	00	0000	0001		K	110TH	00	0000	0001		2644
1022	00	0000	0002		K	21XXX	00	0000	0002		2646
0774	00	0000	0004		K	41XXX	00	0000	0004		2648
0068	00	0000	0008		K	810TH	00	0000	0008		2650
0202	00	0000	0025		K	251XX	00	0000	0025		2652
1119	00	0000	0029		K	291XX	00	0000	0029		2654
0320	00	0000	0050		K	501XX	00	0000	0050		2656
0722	00	0000	0080		K	89THX	00	0000	0080		2658
1308	00	0000	0090		K	99THX	00	0000	0090		2660
0672	00	0000	0800		K	88THX	00	0000	0800		2662
0409	00	0000	2000		K	2000I	00	0000	2000		2664
0622	00	0000	8000		K	87THX	00	0000	8000		2666
0038	00	0001	0000		K	16THX	00	0001	0000		2668
0492	00	0002	0000		K	20XXX	00	0002	0000		2670
0132	01	0000	0000		K	12NOX	01	0000	0000		2672
0254	10	0000	0000		K	11STX	10	0000	0000		2674
1044	80	0000	0000		K	81STX	80	0000	0000		2676
0114	90	0000	0000		K	91STX	90	0000	0000		2678
0395	00	0210	0000		K	ZTABL	00	0210	0000	LENGTH OF SYMBOL TABLE	2680
					PAT						2682

INSTRUCTIONS LISTED IN ORDER OF DATA ADDRESS

0520	00	0000	0329	R	X0003	00	0000	X1	DRUM, OLD SYMB, REG. FIXED	704
0523	00	0000	0519	R	X0006	00	0000	X0002	CORE, TREAT SAME AS BLANK	708
0524	00	0000	0329	R	X0007	00	0000	X1	OTHER ADDRESS, FIXED	710
0527	00	0000	0081	R	X0010	00	0000	F	OLD SYMBOL, ORUM, REGION, FIXED	728
0264	45	0000	0001	P	C4	45	0000	0001	DIFFERENCE OF BRINGING AND STORING ORD	1130
0277	00	0000	0269	R	M0010	00	0000	M0002	NEW SYMBOL, BUT SYMBOL TABLE FULL	1216
0624	00	0000	0376	J		00	0000	A	TAG-IDENTIFICATION, AND EXIT FROM 18	1246
0727	00	0000	0631	J		00	0000		ADDEND AND EXIT FOR SUBROUTINE 18	1264
0927	00	0000	0919	R	C0010	00	0000	C0002	NEW SYMBOL, BUT TABLE FULL, N.G.	1284
0373	01	0000	0373	R	J0006	01	0000	J0006	800X RANGE, ERROR.	1318
0378	60	0000	0605			60	0000		BRING APPROPRIATE TABULAR VALUE	1568
0687	00	0000	0991	J	F	00	0000		TO BUILD EXIT FROM SUBROUTINE 13	1622
0428	65	0000	0756	J		65	0000		BRING ADDENDS	1842
0423	00	0000	0001	R	L0006	00	0000	SETCC	800X ADDRESS, QUIT IMMEDIATELY	2102
0425	00	0000	0419	R	L0008	00	0000	L0002	MISCELLANEOUS ADDRESS, QUIT AND OMIT	2110
1226	00	0000	0476	J		00	0000	00008	ADDEND AND EXIT FOR USE IN SUBR 18	2138
0928	00	0000	0932	J		00	0000		ADDEND AND EXIT FOR SUBROUTINE 18	2154
0478	00	0000	0470	R	00010	00	0000	00002	SYMB TABLE FULL OR DRUM PACKED, QUIT	2172
0581	00	0000	0573	R	10010	00	0000	10002	SYMBOL TABLE FULL, OMIT PUNCHING ADDR	2232
0982	00	0000	1336	J		00	0000		ADDEND AND EXIT FOR INDEXING	2274
1090	11	0000	0001	P	K	11	0000	0001	TO CHANGE ADDING ORDER TO STORING ONE	2514
1232	69	0001	1025	J	ZQ	69	SETCC	SUB2R	TO RESERVING SUBROUTINE	2444
1333	69	0001	1075	J		69	SETCC	SUB2U	TO UNRESERVING SUBROUTINE	2452
1118	44	0001	0934			44	SETCC		0, QUIT BECAUSE WE HAVE FOUND LONG SYM	2508
1218	11	0001	0001	P	KA	11	0001	0001	TO CHANGE STORING ORDER TO ADDING ONE	2516
0702	00	0001	0001	P	C6	00	0001	0001	TO MODIFY VARIABLE BRINGING ORDER	2602
0027	69	0001	0006	R	0027	69	SETCC	SUBR1	TO INITIALIZING SUBROUTINE	2638
0568	00	0001	0375	J		00	0001	J0008	ADDEND AND EXIT FOR SUBROUTINE 18	1312
1028	00	0001	0577	J		00	0001	I0006	ADDENDS AND EXIT FOR USE IN SUBR 18.	2206
1331	00	0001	1236	J		00	0001		ADDEND AND EXIT FOR SUBROUTINE 18.	2216
1032	00	0001	1288	J		00	0001		ADDEND AND EXIT FOR INDEXING	2294
1099	69	0001	1025			69	SETCC	SUB2R	RESERVE AND PUNCH CARO	2358
0994	60	0003	0057			60	READC	SB10A	PUNCH	2544
0100	69	0003	0006	N	0100	69	READC	SUBR1	INITIALIZE AT START OF ASSEMBLY	362
0015	60	0003	0057	J		60	REAOC	SB10A	PREPARE TO PUNCH CARO	392
0001	60	0003	0709	N	SETCC	60	READC	SB10	SEQUEL TO MANY PSEUDO-OPS.	1580
1338	60	0003	0057	C	ALL	60	READC	SB10A	PREPARE TO PUNCH AND READ NEXT CARO	2304
1233	10	0003	0709	N	BP	10	READC	SUB10	BY-PASS INVALID CARO	2360
0051	69	0004	0107	N	MOFLP	69	P	A	STARTED SEARCH WITH MOF OPERATION	398
0256	69	0009	0039			69		SUB14	GET OP, C.I., TAGS, OPTIM, ALOPT.	660
0073	60	0010	0709	R	0073	60	0010	SUB10	PUNCH USUAL DUMMY OUTPUT CARD	2550
0007	44	0011	0012			44	MDFLI	ABCOE	IF SO, START FORWARD SEARCH	388
0283	96	0012	0138			96	ABCOE		IF WITH I, PROCESS LIKE 08	538
0141	69	0012	0045	J		69	ABCOE	SUB20	PROCESS D, AND TO 08 ROUTINE	542
0011	69	0014	0107	N	MDFLI	69	I	A	STARTED SEARCH WITH INDEXED D	400
0012	69	0015	0018	C	ABCDE	69		SUB21	PROCESS I	390
0321	10	0024	8003			10		8003	STORE PROCESSED CARO I	878
0023	45	0026	0077			45	QUIT		0, IT IS NOT, AND WE QUIT	436
0071	44	0026	0076			44	QUIT		IF SO, QUIT SEARCH I	444
0060	44	0026	0064	J		44	QUIT		IF N G, QUIT SEARCH	456
0150	94	0026	0055			94	QUIT		IF SO, QUIT SEARCH	462
0307	45	0026	0136			45	QUIT	B	IF SO, QUIT SEARCH	466
0173	44	0026	0078			44	QUIT		IF I IS N G, QUIT SEARCH	474
0457	45	0026	0111			45	QUIT		IF SO, QUIT SEARCH	486
0734	46	0029	0940			46	0029		IF SPEC CHAR, TO BLR DIRECTLY	2484
0028	11	0031	0035	X		11	O		STORE SET OF DATA I	414
0078	45	0032	0019			45	BCOEF	BACKW	IF I IS FIXED, START BACKWARD	476
0336	95	0032	0012			95	BCDEF	ABCDE	WITH I, EXIT ACCORDING TO WHY SEARCH	612
0081	91	0034	0436	C	F	91	XX8		MULTIPLE BRANCH ACCORDING TO TAGS I	738
0033	69	0036	0039			69		SUB14	OP, C.I., OPTIM, ALOPT, ETC.	376
0585	69	0038	0741			69	16THX		PRESET T AS POSITIVE 1 I	1502
0686	66	0038	0643	N	SHRT	66	16THX		SHORT, PRESET T AS -1 I	1510
0462	16	0038	0793			16	16THX	ST	COMPARISON CONSTANT FOR LAST PART	1642
0562	15	0038	0793			15	16THX	ST	COMPARISON CONSTANT FOR LAST PART	1666
1034	16	0038	0744	A		16	16THX		DUPLICATE INTO AVAILABILITY TABLE I	2410
0035	15	0038	0043			15	16THX		STORE SET OF DATA I	416
0161	15	0038	0143			15	16THX		BRING BACK A SET I	560
0327	16	0038	0243			16	16THX	LOOP	BRING BACK A SET I	642
0265	16	0038	0293			16	16THX		BRING BACK A SET I	652
0259	16	0038	0443			16	16THX		STORE PROCESSED CARD I	886
0547	15	0038	8002	C	A	15	16THX	8002	BRING BACK A SET OF RESULTS I	922
0087	24	0040	0193	0040		24	SAVED		SAVEO SAVE D FROM CARD THAT STARTED SRCH I	628

0135	69	0040	0343		69	SAVE0		BRING BACK L AND D)	678
0041	10	0044	8003	C LOOP	10		8003	STORE SET OF DATA)	408
0147	10	0050	8003		10	QS	8003	STORE SET OF DATA)	420
0048	94	0051	0053		94	MDFLP		D, YES	384
0049	24	0052	0205	0052	24	DONE		PRESET EXIT FROM BACKWARDS ROUTINE)	632
0090	24	0052	0468	0052	24	DONE	NO	WITH L. ALTER EXIT)	684
0397	69	0052	0305	J LNM	69	DONE	SU8R4	PROCESS L BACKWARDS	762
0102	65	0052	0201	J LN	65	DONE	SU822	SAVE ORCEQ	786
0257	69	0060	0061		69		SU8R3	IF SO, IS IT FIXED)	454
0109	65	0062	0217	J	65	OPREG		SAVE DYNAMIC LEVEL OF L FROM D)	818
0406	65	0062	0267	J	65	OPREG		WHICH L IS LESS, MEASURED ON CIRCLE)	834
0376	20	0062	0925	0062	N A	20	OPREG	STORE ADDRESS FOR OPTIMIZING NEXT ADDR	1278
0371	24	0062	0615	0062	N J0004	24	OPREG	DRUM OR DRUM EQUIVALENT	1308
0335	20	0062	0177	0062	N S810C	20	OPREG	FROM 4-5, 19-20. STORE DYNAMIC LEVEL	1602
1320	65	0062	0967		65	OPREG		WHICH ONE DO WE USE)	1886
0956	15	0062	1017		15	OPREG		ADD OLD DYNAMIC LEVEL	1902
0163	69	0066	0400	J	69		SU8R5	PROCESS BACKWARDS D ONCE MORE	852
0065	16	0068	0023		16	810TH		IS CARD OF TYPE 08)	434
0165	16	0068	0323		16	810TH		IS IT TYPE 08)	496
0218	60	0068	0177	R F0001	60	810TH	EXITX	ADDRESS IS N G	1154
0219	65	0068	0177	R F0002	65	810TH	EXITX	BLANK ADDRESS	1156
0221	65	0068	0177	R F0004	65	810TH	EXITX	SYMBOLIC ADDRESS, WITH UNDEFINED SYMB	1160
0223	65	0068	0177	R F0006	65	810TH	EXITX	CORE ADDRESS	1164
0224	60	0068	0177	R F0007	60	810TH	EXITX	OTHER ADDRESS. USUALLY PART OF CONST.	1166
0215	22	0069	0022	0069	22	OC	OC	NOT 1ST CARD. MAKE COMP. CONST.)	548
0106	16	0069	0623		16	QC		IS THIS LAST SET	568
0571	69	0074	0377		69	BRNG		MAKE NEW BRINGING ORDER)	906
0522	15	0075	0329	R X0005	15	13RD	X1	800X. CALL IT UNFIXED	696
0518	15	0075	0329	R X0001	15	13RD	X1	ADDRESS N G. CALL UNFIXED	698
0521	15	0075	0329	R X0004	15	13RD	X1	NEW SYMBOL. UNFIXED	706
0176	15	0079	0233		15	710		IS TYPE 01	500
0229	10	0082	8002	C LOOP	10	SEND	8002	BRING BACK A SET)	552
0083	65	0086	0041		65	SEND	LOOP	STORE SET OF DATA)	406
0076	45	0088	0026		45	C	QUITT	IF SO, QUIT SEARCH)	446
0036	69	0089	0042	J	69		SU819	PROCESS L	378
0088	69	0091	0039	C C	69		SU814	GET OP, C.I., TAGS, OPTIM, ALOPT.	448
0089	69	0092	0045	J	69		SUB20	PROCESS O	380
0092	69	0095	0048	J	69	OPTIM		IS IT AN MOF OPERATION)	382
0688	65	0095	0699	N FD	65	OPTIM		FORWARD D. L-D ADDENDS, LEFT END LOWE	1786
0490	65	0095	0749	N F18D	65	OPTIM		FORWARD I OR BACKWARD D. GET ADDENDS	1798
0706	69	0095	0548		69	OPTIM		OPTIMIZING TAGS	1826
1222	16	0095	0785		16	OPTIM	SEO	REDUCE ADDENDS BY 1 FOR 80+82, OR 88	1848
1306	24	0095	0648	0095	24	OPTIM	OPTIM	STORE OPTIMIZING ADDENDS AND TAGS	1966
0471	69	0095	0748	R D0003	69	OPTIM		BLANK D. IS IT MDF OPERATION	2128
1049	69	0095	0948	C 8	69	OPTIM		SEVERAL CASES. WHAT KIND OF OP	2166
0120	20	0095	1098	0095	N Q0003	20	E	DRUM. PUT I-EQUIVALENT INTO E	2324
0122	20	0095	1098	0095	R Q0005	20	E	800X ADDRESS	2328
0123	20	0095	1098	0095	R Q0006	20	E	CORE ADDRESS	2330
0124	20	0095	1098	0095	R Q0007	20	E	OTHER ADDRESS	2332
1267	65	0095	0127	J	65	E	Q0010	BRING VALUE OF I INTO LOWER	2338
0594	60	0095	1099	J	60	E		BRING SYMBOL-EQUIVALENT TO UPPER	2356
0193	69	0096	0049		69	DN1		PRESET EXIT FROM BACKWARDS ROUTINE)	630
0298	69	0101	0018	J	69		SU821	PROCESS I FORWARDS	774
0149	69	0102	0018	J	69	LN	SUB21	PROCESS I FORWARDS	784
0195	69	0102	0045	J	69	LN	SU820	PROCESS D FORWARDS	794
0148	20	0103	0056	D103	N OUT	20	FINAL	FINAL STORE VARIABLE ORDER	422
0032	65	0103	0407	N BCDEF	65	FINAL		IS STORAGE AREA FULL)	478
0227	65	0103	0041	C LP	65	FINAL	LOOP	ALL CASES, BACK TO START OF LOOP	512
0258	65	0103	0557	N SKP	65	FINAL		NOT 1ST CARD. MAKE COMP. CONST.)	544
0205	65	0103	0757		65	FINAL		BRING BACK A SET)	634
0456	60	0108	0263		60	SA		ARE WE TO LAST LINE OF TABLE)	1018
0331	21	0108	0361	D108	21	SA	SA	STORE VARIABLE STORING ORDER	1026
0312	60	0108	0513		60	SA		MODIFY TO TAKE FIRST LINE, SAME COLUMN	1120
0362	60	0108	0563		60	SA		LAST LINE AND END OF WORD. MODIFY)	1120
0747	69	0108	80D2		69	HSYMB	8002	STORE SYMBOL)	1334
1005	11	0108	0763		11	HSYMB		SOME SYMBOL WAS FOUND. SUBTRACT OURS	1422
0636	21	0108	0961	D108	N ABC	21	HSYMB	ADDRESS NOT BLANK. STORE SYMBOL	1440

0416 60 0108 1013	N SYM	60 HSYMB		SYMBOLIC ADDRESS. BRING SYMBOL	1496
0356 65 0109 0113	J	65	SSB	BRING BACK BLANB AND ORCEB	816
0357 69 0110 0061		69	SUBR3	IS I A FIXED ADDRESS)	470
0791 01 0111 1219	J FULL	01 0111	NG	STOP IF SYMBOL TABLE IS FULL	1556
0309 46 0112 0213		46	OUT	CLEAR REGION AND SYMBOL TABLES)	958
0111 69 0114 0017		69 91STX		CHANGE FIRST-CARO INOICATOR TO 2ND)	488
0188 69 0114 0067	N B	69 91STX		START ON L. FIRST-CARO TAG)	614
1109 69 0114 0617		69 91STX		CHANGE DRUM TAG TO 9)	1700
0016 65 0114 0984	R 0016	65 91STX		TAG IN CASE L IS UNDEFINED SYMBOL)	2522
0263 11 0116 0671		11 796		ARE WE TO LAST LINE OF TABLE)	1020
0164 24 0117 0020 0117		24 VAR2X		VAR2X PRESET TO STORE AFTER PROCESSING)	620
0096 65 0117 0321	J DN1	63 VAR2X		ALL CASES. STORE PROCESSED CARD)	876
0398 20 0117 0570 0117	N OUT2	20 VAR2X		VAR2X STORE PROCESSED CARD)	892
0137 65 0117 0571	J ENOED	65 VAR2X		MAKE NEW BRINGING ORDER)	904
0987 15 0132 1037		15 12NOX		SHOULD WE INTERCHANGE ADDENDS)	1870
0529 15 0132 0187	R X0012	15 12NOX	X2	800X ADDRESS. UNFIXED	720
0525 15 0132 0187	R X0008	15 12NDX	X2	0 IS N G. CALL UNFIXED	722
0528 15 0132 0187	R X0011	15 12NOX	X2	NEW SYMBOL. UNFIXED	730
0301 15 0132 0237		15 12NOX		MODIFY TO USE ROUTINE FOR UNFIXED D)	826
0351 16 0132 0287		16 12NDX	XY	MODIFY ALOPT AGAIN)	856
0231 61 0134 8003		61	8003	CLEAR REGION AND SYMBOL TABLES)	952
0133 96 0136 0088		96 B	C	WITH I OR L, RESPECTIVELY	430
0288 69 0137 0090	N L	69 ENOED		WITH L. ALTER EXIT)	682
0286 69 0139 0018	J JUMP	69	SUB21	PROCESS I	586
0436 99 0140 0291		99	MISC	MULTIPLE BRANCH ACCORDING TO TAGS)	740
0138 69 0141 0042		69	SUB19	PROCESS L	540
0091 95 0144 0046	J	95 K		IS O ACTUALLY AN ADDRESS	450
0064 45 0144 0019		45 K	BACKW	IF FIXEO, START BACKWARD PROCESSING	458
0243 10 0146 8002	N LOOP	10	8002	BRING BACK A SET)	646
0043 44 0147 0148		44	OUT	STORE SET OF DATA)	418
0196 69 0149 0305	J	69	SUBR4	PROCESS L BACKWARDS	782
0037 24 0151 0104 0151		24 R0001		R0001 IF SO, PUT IN DUMMY MOVEABLE WORDS)	504
0250 24 0150 0408 0150	Q QS	24 R0000	X	R0000 RESTORE COMPARISON CONSTANT AND MODIFY	750
0044 69 0151 8002	J	69 R0001	8002	STORE SET OF DATA)	410
0077 60 0151 0005		60 R0001		ALPHABETIC L	438
0253 70 0151 1007		70 R0001		READ ANOTHER CARO	492
0082 24 0151 0204 0151	J SEND	24 R0001	X	R0001 BRING BACK A SET)	556
0261 24 0151 0408 0151	Q Q	24 R0001	X	R0001 COMPARISON CONSTANT FOR END OF SET	748
0080 60 0151 0455		60 R0001		ALPHABETICAL L INTO UPPER	1174
0480 60 0151 0712		60 R0001		ALPHABETICAL L TO UPPER	2070
1212 10 0151 1262		10 R0001		ALPHABETIC L	2252
1329 70 0151 1102	N OUT	70 R0001		WHEN DONE, READ NEXT CARO	2400
0046 60 0152 0257		60 R0002		IF SO, IS IT FIXEO)	452
0104 24 0152 0105 0152		24 R0002		IF SO, PUT IN DUMMY MOVEABLE WORDS)	506
0350 60 0152 1057		60 R0002		ALPHABETIC D, TO SEE WHETHER FIXED	714
0230 60 0152 0958		60 R0002		ALPHABETIC O INTO UPPER	1224
0630 60 0152 0762		60 R0002		ALPHABETICAL O INTO UPPER	2118
0574 60 0152 1012	R I0003	60 R0002		BLANK I. IS O ALSO BLANK	2190
0614 10 0152 1312		10 R0002		ALPHABETIC O	2266
1098 60 0152 0914	C X	60 R0002		ALPHABETIC O	2334
0097 65 0152 1116	R 0097	65 R0002		REGION DESIGNATOR TO RIGHT END)	2474
0944 65 0152 1318	J	65 R0002		ALPHABETIC SYMBOL	2534
0093 65 0152 1345	R 0093	65 R0002		ALPHABETIC O	2620
0136 60 0153 0357	C B	60 R0003		IS I A FIXEO ADDRESS)	468
0105 24 0153 0236 0153		24 R0003	YYY	R0003 IF SO, PUT IN DUMMY MOVEABLE WORDS)	508
0255 60 0153 0957		60 R0003		BRING ALPHABETIC I	690
0330 60 0153 1108		60 R0003		ALPHABETIC I TO UPPER	1292
0680 60 0153 0962		60 R0003		ALPHABETIC I TO UPPER	2180
0664 10 0153 0714		10 R0003		ALPHABETIC I	2286
0054 60 0153 0764	R 0054	60 R0003	BOTH	EQU ENTRY. ALPHABETIC I	2312
0025 60 0153 0764	R 0025	60 R0003	BOTH	SYN ENTRY. ALPHABETIC I	2314
1018 60 0153 1214	J	60 R0003		IS THE I-ADDRESS BLANK	2384
1282 65 0153 1016		65 R0003		ALPHABETICAL I	2460
0696 65 0153 0746		65 R0003		ALPHABETIC I	2628
0941 21 0154 0759 0154		21 R0004		R0004 TO SUPPRESS IN CASE WE SEARCH	1998
0430 69 0154 1110		69 R0004		CONTROL INFORMATION FOR PUNCHING)	1926
1350 69 0154 1268		69 R0004		CONTROL INFORMATION)	2528

0236	24	0155	0227	0155	C YYY	24	R0005	LP	R0005	EITHER CASE, SOMETHING INTO R0005	510
0031	69	0155	8002		O O	69	R0005	8002		COMPARISON CONSTANT FOR STORING	514
0607	24	0155	0204	0155	O O	24	R0005	X	R0005	COMPARISON CONSTANT FOR BRINGING SET	596
0146	24	0155	0408	0155	J	24	R0005	X	R0005	BRING BACK A SET)	648
0989	65	0155	1210			65	R0005			BRING TAGS AND OPERATION	1930
0050	69	0156	8002		O OS	69	R0006	8002		TO RESTORE COMPARISON CONST + ADVANCE	516
0200	24	0156	0204	0156	O OS	24	R0006	X	R0006	TO RESTORE AND ADVANCE	598
1112	65	0157	1212		J	65	R0007			NUMERICAL L	2250
1286	65	0158	0614		C PROO	65	R0008			NUMERICAL O	2264
0094	60	0158	1114		R 0094	60	R0008			BEGINNING OF REGION TO BE RESERVED	2380
1068	60	0158	1264			60	R0008			BRING FIRST ADDRESS AGAIN	2388
0730	21	0158	1314	0158		21	R0008		R0008	STORE ADVANCED O ADDRESS	2392
1066	16	0158	1332			16	R0008	BL		SUBTRACT NUMERICAL O FROM LOWER	2466
1332	10	0158	0179		C BL	10	R0008	EXITZ		NUMERICAL O INTO UPPER	2468
0784	69	0158	8002			69	R0008	8002		ADDRESS OF 0001 OF REGION	2490
0596	65	0158	0646			65	R0008	SDO		BRING NUMERICAL O	2624
1238	65	0159	0664		C PROI	65	R0009			NUMERICAL I	2284
0098	65	0159	0964		R 0098	65	R0009			REG ENTRY. NUMERICAL I	2362
1314	11	0159	0716			11	R0009			SUBTRACT END OF AREA TO BE RESERVED	2394
1330	65	0159	1066			65	R0009			NUMERICAL I INTO LOWER	2464
0796	65	0159	0946			65	R0009	SCO		BRING NUMERICAL I	2632
0058	65	0160	0065		N NO	65	R0010			IS CARO OF TYPE 08)	432
1007	65	0160	0165			65	R0010			IS IT TYPE 08)	494
1102	65	0160	0766			65	R0010			IS THIS ANOTHER RBO CARO)	2402
0794	60	0160	0709			60	R0010	SUB10		PUNCH CARO AND PROCEED	2432
0003	70	0161	1041		REAOC	70	R0011			READ ONE CARO	364
0508	45	0162	0178			45		EXITY		O, WE HAVE NOT FINISHED. I, WE HAVE.	1012
0560	65	0163	0113			65		SSB		BRING BACK BLANK AND ALOFT ONCE MORE	850
0313	46	0166	0178			46		EXITY		I, OVER 1999. WE RESERVE NOTHING	1046
0013	22	0167	0070			22	XXXX1			XXXX1 BRING BACK A SET)	638
0377	22	0167	0620			22	XXXX1	LOOP		MAKE NEW BRINGING ORDER)	908
0673	65	0167	0621	J		65	XXXX1			BRING BACK A SET OF RESULTS)	912
0448	20	0167	0620	0167	N OUT	20	XXXX1	LOOP		XXXX1 STORE MODIFIED BRINGING ORDER + REPEAT	934
0555	20	0167	0670	0167		20	XXXX1			XXXX1 MAKE BRINGING ORDER)	1348
0713	21	0167	0720	0167	C RES	21	XXXX1			XXXX1 STORE ADDRESS TEMPORARILY	1404
1273	65	0167	0177	J		65	XXXX1	EXITX		EQUIVALENT BACK TO LOWER AND EXIT	1408
0390	22	0167	0441	0167		22	XXXX1	SUB13		XXXX1 PREPARE EXIT AND GO TO SUBROUTINE 13)	1620
1209	22	0167	8001	0167		22	XXXX1	8001		XXXX1 NEW AVAILABILITY WORD INTO ORUM)	1736
0935	20	0167	1320	0167	C P	20	XXXX1			XXXX1 ALL CASES. STORE ADOENOS	1884
1014	65	0167	0771			65	XXXX1			000. SEPARATE LAST ADOENO)	1892
0414	65	0167	0985		N EVN	65	XXXX1	BOTH		EVEN. WE SHOULD USE FIRST ADOENO	1898
1261	23	0167	0971	0167		23	XXXX1			XXXX1 STORE LAST 4 DIGITS OF OPTIM	1970
1089	10	0167	1071		C AX	10	XXXX1			ADD IN TAGS FROM OPTIM	1980
1281	20	0167	1121	0167		20	XXXX1			XXXX1 STORE ADDRESS	2034
1221	10	0167	1271			10	XXXX1			INDEXABLE ADDRESS INTO UPPER	2048
0999	15	0167	0178		C A	15	XXXX1	EXITY		ADD ADDRESS BEING INDEXED. EXIT.	2060
0621	16	0174	0379			16	OL			BRING BACK A SET OF RESULTS)	914
0072	65	0175	0429			65	AVAL			MAKE DRUM AVAILABLE)	968
0323	45	0176	0227			45		LP		O, IT IS NOT. I, IT IS.	498
0061	24	0177	0030	0177	N SUBR3	24	EXITX			EXITX STORE EXIT	1148
0305	24	0177	0080	0177	N SUBR4	24	EXITX			EXITX STORE EXIT	1172
0400	24	0177	0230	0177	N SUBR5	24	EXITX			EXITX STORE EXIT	1222
0142	24	0177	0330	0177	N SUBR6	24	EXITX			EXITX STORE EXIT	1290
1024	24	0177	0380	0177	N SUBR8	24	EXITX			EXITX STORE EXIT	1374
0434	71	0177	8003			71	P0001	8003		PUNCH CARO. PERFORM UPPER.	1594
0039	24	0177	0430	0177	N SUB14	24	EXITX			EXITX STORE EXIT	1924
0042	24	0177	0480	0177	N SUB19	24	EXITX			EXITX STORE EXIT	2068
0045	24	0177	0630	0177	N SUB20	24	EXITX			EXITX STORE EXIT	2116
0948	91	0177	0335			91	EXITX	SB10C		O, BRANCH. I, ARITHMETIC.	2168
0018	24	0177	0680	0177	N SUB21	24	EXITX			EXITX STORE EXIT	2178
0988	69	0177	0661			69	EXITX	SUBR7		STORE SYMBOL AND ITS EQUIVALENT	2230
1249	20	0177	1334	0177	C LOOP	20	P0001		P0001	STORE LOCATION OF AVAILABILITY WORD	2560
0172	71	0177	1096		R T0005	71	P0001			PUNCH A CARO OF THE TABLE	2590
0006	24	0178	0231	0178	N SUBR1	24	EXITX			EXITX STORE EXIT	950
0415	46	0178	0569			46	EXITX			MAKE DRUM AVAILABLE)	976
1025	24	0178	0381	0178	N SUBR2R	24	EXITX			EXITX ENTRY TO RESERVE. STORE EXIT.	1030
1075	24	0178	0431	0178	N SUB2U	24	EXITX			EXITX ENTRY TO UNRESERVE. STORE EXIT.	1034
0976	20	0178	0931	0178		20	EXITX			EXITX STORE EXIT OUT OF LOWER	1612
0693	69	0178	0981		C SSW	69	EXITX			ALL CASES. GOING WHICH DIRECTION	1630
1122	65	0178	1077			65	EXITX	PAKT		MODIFY EXIT)	1704
0586	24	0178	1231	0178	N SUB17	24	EXITX			EXITX STORE EXIT INSTRUCTION	1992
0677	24	0178	1281	0178	N INOEX	24	EXITX			EXITX STORE EXIT	2032
1121	65	0178	0783			65	EXITX			O-POSITION HOLOS O FOR O, 1 FOR I)	2036
0168	24	0178	0945	0178	R T0001	24	P0002		P0002	STORE AVAILABILITY WORD	2574
0661	24	0179	0432	0179	N SUBR7	24	EXITZ			EXITZ STORE EXIT	1328

0063	24	0179	0482	0179	N	SUBR9	24	EXITZ		EXITZ	STORE EXIT	1434
1069	10	0179	8003		C	DON	10	EXITZ	8003		FROM MANY PLACES. MULTIBRANCH	1574
0441	24	0179	0532	0179	N	SUB13	24	EXITZ		EXITZ	STORE EXIT	1778
0785	69	0179	0582		C	SEO	69	EXITZ			ALL CASES. WHICH DIRECTION	1850
1325	69	0179	0632				69	EXITZ			IF BACKWARD L, SUBTRACT TAG-COUNT)	1906
0740	24	0179	1282	0179	N	SUB	24	EXITZ		EXITZ	STORE EXIT	2458
1299	20	0179	0745	0179			20	P0003		P0003	STORE LOCATION OF AVAILABILITY WORD	2564
0169	24	0180	0995	0180	R	T0002	24	P0004		P0004	STORE AVAILABILITY WORD	2578
0057	15	0181	0635		C	SB10A	15	P0005			ADVANCE CARO NUMBER)	1588
1027	20	0181	0434	0181			20	P0005		P0005	ADVANCE CARO NUMBER)	1592
0650	69	0181	8002		O	OS2	69	P0005	8002		RESTORE CONSTANT AND MODIFY	898
0213	20	0181	0234	0181	N	OUT	20	P0005		P0005	CLEAR CARO NUMBER	962
0274	20	0182	0385	0182	R	M0007	20	P0006		P0006	CORE ADDRESS. STORE FOR PUNCHING	1200
0421	20	0182	1285	0182	N	L0004	20	P0006		P0006	ORUM. STORE ADDRESS FOR PUNCHING	2086
0595	65	0182	1337				65	P0006	MW		BRING BACK NUMERICAL ADDRESS	2092
0424	20	0182	1335	0182	R	L0007	20	P0006		P0006	CORE ADDRESS. STORE FOR PUNCHING	2104
1324	20	0182	1286	0182			20	P0006	PROD		STORE L FOR PUNCHING	2258
0170	24	0182	1045	0182	R	T0003	24	P0006		P0006	STORE AVAILABILITY WORD	2582
0020	69	0182	0085				69	P0006			SAVE L FROM CARO THAT STARTED SRCH)	622
0241	24	0182	0135	0182			24	P0006		P0006	BRING BACK L AND D)	676
0355	69	0182	8002		O	O2	69	P0006	8002		COMPARISON CONSTANT FOR END OF SET	896
0242	24	0182	0285	0182	J	SEND	24	P0006	x	P0006	BRING BACK A SET OF RESULTS)	926
0271	20	0182	0335	0182	N	M0004	20	P0006	SB10C	P0006	ORUM ADDRESS. STORE FOR PUNCHING	1192
0566	22	0183	0986	0183			22	P0007		P0007	STORE OPERATION FOR PUNCHING	1934
0598	60	0183	1237				60	P0007			OPERATION IN 0 OF UPPER	1960
1340	24	0183	0994	0183			24	P0007		P0007	STORE FIRST 2 DIGITS IN OP POSITION	2542
0652	20	0183	8003	0183			20	P0007	8003	P0007	STORE LOCATION OF AVAILABILITY WORD	2572
0631	20	0184	0487	0184			20	P0008		P0008	STORE CORE ADDRESS FOR PUNCHING	1266
0925	20	0184	0177	0184	R	C0008	20	P0008	EXITX	P0008	MISC ADDR. STORE FOR PUNCHING. EXIT	1280
0402	60	0184	0789		N	SHOP	60	P0008			SHIFT OPERATION. FORWARD I, BACKWARD L.	1816
0660	60	0184	0939		N	XAS	60	P0008			INDEXING OP, FORWARD I, BACKWARD L.	1832
0992	24	0184	1287	0184			24	P0008		P0008	STORE THE ADDRESS FOR PUNCHING	2006
0932	20	0184	0788	0184			20	P0008		P0008	STORE THE CORE ADDRESS FOR PUNCHING	2156
0476	20	0184	1049	0184	R	D0008	20	P0008	B	P0008	MISC ADDRESS. USUALLY A CONSTANT	2164
1336	20	0184	1238	0184			20	P0008	PROI	P0008	STORE D FOR PUNCHING	2276
0602	20	0184	1340	0184			20	P0008		P0008	STORE NEXT 4 DIGITS IN D POSITION	2540
0171	24	0184	1095	0184	R	T0004	24	P0008		P0008	STORE AVAILABILITY WORD	2586
0191	69	0184	0087				69	P0008			SAVE D FROM CARD THAT STARTED SRCH)	626
0343	24	0184	0137	0184			24	P0008	ENDED	P0008	BRING BACK L AND D)	680
0375	20	0185	0177	0185	R	J0008	20	P0009	EXITX	P0009	OTHER ADDRESS. USUALLY A CONSTANT	1322
0577	20	0185	0177	0185	R	I0006	20	P0009	EXITX	P0009	800X ADDRESS. STORE AND EXIT	2212
1236	20	0185	0938	0185			20	P0009		P0009	STORE CORE ADDRESS FOR PUNCHING	2218
0579	20	0185	0177	0185	R	I0008	20	P0009	EXITX	P0009	OTHER ADDRESS. STORE AND EXIT.	2226
0580	20	0185	0988	0185	N	I0009	20	P0009		P0009	STORE EQUIVALENT OF NEW SYMBOL	2228
1288	20	0185	1338	0185			20	P0009	ALL	P0009	STORE I FOR PUNCHING	2296
1318	24	0185	1290	0185			24	P0009		P0009	STORE LAST 4 DIGITS OF CONSTANT IN I	2536
1349	20	0185	0795	0185			20	P0009		P0009	STORE LOCATION OF AVAILABILITY WORD	2568
0499	20	0186	0689	0186			20	P0010		P0010	C.I. TO PUNCH X-9)	1584
0627	10	0186	0941		N	SB10B	10	P0010			FROM 4-6,19-21. TO SUPPRESS)	1596
0759	21	0186	0177	0186			21	P0010	EXITX	P0010	FROM 4-6,19-21. TO SUPPRESS)	1600
1110	24	0186	0989	0186			24	P0010		P0010	CONTROL INFORMATION FOR PUNCHING)	1928
1321	10	0186	1343		N	NGL	10	P0010			FIX C.I. SO THAT L WILL NOT PUNCH)	2260
1343	21	0186	1286	0186			21	P0010	PROD	P0010	FIX C.I. SO THAT L WILL NOT PUNCH)	2262
1083	10	0186	0494				10	P0010			FIX C.I. SO THAT D WILL NOT PUNCH)	2280
0494	21	0186	1238	0186			21	P0010	PROI	P0010	FIX C.I. SO THAT D WILL NOT PUNCH)	2282
1239	10	0186	0544				10	P0010			FIX C.I. SO THAT I WILL NOT PUNCH)	2300
0544	21	0186	1338	0186			21	P0010	ALL	P0010	FIX C.I. SO THAT I WILL NOT PUNCH)	2302
1268	24	0186	1240	0186			24	P0010		P0010	CONTROL INFORMATION)	2530
1244	24	0186	1294	0186			24	P0010		P0010	C.I. FOR AVAILABILITY TABLE)	2554
0024	69	0186	8002		J		69	P0010	8002		STORE PROCESSED CARD)	880
0488	24	0186	0285	0186	O	O	24	P0010	x	P0010		940
0703	90	0187	0528				90	X2	X0011		NO, FIXED. YES, UNFIXED.	726
0700	24	0187	0285	0187	O	OS	24	P0011	x	P0011		942
0386	69	0189	0142				69		SUBR6		PROCESS I BACKWARD	672
0487	69	0190	0441				69		SUB13		GET DYNAMIC LEVEL OF CORE ADDRESS	1268
0239	69	0192	0142		N	9X8	69		SUBR6		PROCESS I BACKWARDS	790
0291	69	0194	0142		N	MISC	69		SUBR6		PROCESS I BACKWARDS	758
0192	69	0195	0305		J		69		SUBR4		PROCESS L BACKWARDS	792
0316	00	0195	0000		P	C8	00	0195	0000		MODIFY BRINGING ORDER TO START NEW COL	1142
0393	69	0196	0400		N	898	69		SUBR5		PROCESS D BACKWARDS	780
0266	00	0196	0000		P	C6	00	0196	0000		TO MODIFY BRINGING ORDER FOR NEXT COL	1138
0144	69	0197	0150		C	K	69	ALOFT			IS IT AN MDF OPERATION	460
0468	69	0197	0300		C	NO	69	ALOFT			IS D AN ACTUAL ADDRESS	686
0460	65	0197	0520		J		65	ALOFT	X0003		MULTIPLE EXIT, ACCORDING TO KINO	694
0329	20	0197	0350	0197	C	X1	20	ALOFT		ALOFT	STORE MODIFIED ALOFT	712
0510	65	0197	0527		J		65	ALOFT	X0010		MULTIPLE EXIT	718

D187 20 D197 D081	D197	N X2	2D ALOPT	F	ALDPT	STORE MODIFIED ALOPT	736
D450 65 D197 D251			65 ALDPT		ALOPT	ALOPT TO LOWER	806
0D59 20 D197 D500	D197		2D ALOPT		ALOPT	STORE MODIFIED ALOPT	810
1094 65 D197 D301			65 ALDPT			MODIFY TO USE ROUTINE FOR UNFIXED D)	824
D237 20 D197 D550	0197		2D ALOPT		ALOPT	MODIFY TO USE ROUTINE FOR UNFIXED D)	828
DD66 65 D197 D351		J	65 ALOPT			MODIFY ALOPT AGAIN)	854
D209 20 D197 D397	D197	J	20 ALOPT	LN	ALOPT	STORE NEW ALOPT, AND JUMP BACK	860
0317 65 D197 D287		J	65 ALOPT	XY		BRING ALOPT, AND JUMP BACK	866
D276 69 D197 D950		N MDD09	69 ALDPT			EQUIV DF NEW SYMBOL,	1210
1247 69 D197 D100			69 ALOPT			SHOULD WE RESERVE	1712
0742 65 D197 D051		N BL	65 ALOPT			BACKWARD L. GET ALOPT	1806
1071 21 D197 D177	0197		21 ALOPT	EXITX	ALOPT	STORE ALL OPTIMIZING TAGS	1982
1084 24 D197 D1350	D197		24 ALDPT		ALOPT	TAG IN CASE L IS UNDEFINED SYMBOL)	2526
D247 1D D200 8DD2			1D QS	8D02		BRING BACK A SET)	564
0199 16 D2D2 1107			16 25IXX	BDT		WHICH L IS LESS, MEASURED DN CIRCLE)	844
D348 15 D202 11D7		N AEB	15 25IXX	BDT		WHICH L IS LESS, MEASURED DN CIRCLE)	846
D453 6D D206 DD57			6D	SB1DA		PUNCH CARD	592
D2D3 90 D2D7 DD58			9D	ND		IS THIS FIRST SET DF SEARCH)	426
DD05 69 D2D8 DD61			69	SUBR3		IS L FIXED ADDRESS)	44D
04D7 16 D210 D115			16 MAX			IS STORAGE AREA FULL)	48D
0358 69 D211 0164		C A	69 SEN			PRESET TD STORE AFTER PROCESSING)	618
D311 65 D214 0113		N 2ND	65	SSB		USE I-COMP. BRING BLANB AND ORCEB	862
D463 45 D216 0417			45	ZP		D, ND	1D98
D7D5 D1 D222 11D9		N FULL	01 D222			STOP RECAUSE DRUM IS PACKED	1698
D623 45 D226 016D			45	ROD1D		IF SO, MULTIBRANCH + EXIT FRDM OUITT	57D
D773 21 D228 D481	D228		21 P		P	MAKE SEVERAL VARIABLE ORDERS)	1D7D
0363 15 D228 D583			15 P			MAKE SEVERAL VARIABLE ORDERS)	1080
D232 24 D235 D388	D235		24 T1		T1	SAVE IN CASE I IS BLANK	8D0
D113 69 D235 D438		N SSB	69 T1			SPEC. SUBR. BRING BACK BLANB + ORCEB	868
D233 45 D236 DD37			45 YYY			D, NO. I, YES.	5D2
DD85 24 D238 0191	D238		24 SAVEL		SAVEL	SAVE L FROM CARD THAT STARTED SRCH)	624
D189 69 D238 D241		J	69 SAVEL			BRING BACK L AND D)	674
0289 1D D242 0547			1D SEND	A		BRING BACK A SET OF RESULTS)	92D
0593 69 D246 D249			69 C1			MAKE SEVERAL VARIABLE ORDERS)	1D84
0973 69 D246 D299			69 C1			MODIFY TO TAKE NEXT COLUMN OF TABLE)	11D2
D417 69 D246 D349		N ZP	69 C1			LAST LINE AND END OF WORD. MODIFY)	1112
D143 44 D247 D198			44	OUT		BRING BACK A SET)	562
D347 1D D25D 8002			10 OS	8D02		BRING BACK A SET)	656
D251 16 D254 D059			16 11STX			CHANGE ITS FIRST DIGIT TO 8	8D8
0287 15 D254 D209		C XY	15 11STX			MODIFY ALOPT AGAIN)	858
038D 11 D254 D559			11 11STX			IS FIRST POSITION BLANK	1376
D557 1D D26D D215			10 BRNG			NOT 1ST CARD. MAKE COMP. CONST.)	546
D4D8 11 D261 D265		X	11 O			BRING BACK A SET)	65D
D361 10 D264 80D3			1D C4	80D3		MAKE VARIABLE BRINGING ORDER	1D28
0513 11 D266 D331			11 C6	SU		MODIFY TO TAKE FIRST LINE, SAME COLUMN	111D
D950 9D D271 0505			9D MDDD4			O, CASE WHERE WE DUPLICATE COMPUTATION	1212
D725 45 D278 0679			45	OUT		ARE LAST FOUR CHARACTERS DIGITS)	1454
D226 69 D279 D039			69	SUB14		GET OP, C.I., TAGS, OPTIM, ALOPT.	572
0479 21 D284 0387	0284		21 XXXXA		XXXXA	STORE INITIAL ADDRESS TO BE RESERVEO	1D40
D411 6D D284 D389		C OK	6D XXXXA			MAKE SEVERALVARIABLE ORDERS)	1D58
0333 96 D286 0657			96 JUMP	NO		D, STARTED WITH I. I, WITH L.	58D
D310 69 D286 0045		J	69 JUMP	SUB20		PROCES O	584
0587 65 D29D 80D1			65 Y	80D1		BRINGING ORDER INTO LOWER	1506
D6D6 16 D29D D345		N UND	16 Y			SYMB NOT IN TABLE. SUBTRACT Y	1518
D972 16 D29D D445		N OEF	16 Y			DEFINED SYMBOL. SUBTRACT Y	153D
014D 98 D291 D145			98 MISC	899		MULTIPLE BRANCH ACCORROING TO TAGS)	742
D339 24 D292 DDD3	0292		24 ORCEO	REAOC		ORCEO FINISHED. SAVE SAVOR AS ORCEQ)	938
1335 65 D292 1337			65 ORCEO	MW		DYNAMIC LEVEL OF LAST CORE ADDRESS	21D6
0798 2D D292 1049	0292		2D ORCEO	B		STORE DYNAMIC LEVEL FOR REFERENCE	2162

0998	20	0292	0177	0292		20	ORCEQ	EXITX	ORCEQ	STORE DYNAMIC LEVEL FOR REFERENCE	2224
1067	69	0292	0645			69	ORCEQ			STORE ORCEQ FOR USE WHEN WE START FORW	2240
0491	15	0294	0335			15	XXXX2	SB10C		ADD TO GIVE MODIFIED DYNAMIC LEFEL	1260
1039	20	0294	0797	0294		20	XXXX2		XXXX2	LOCATION OF EQUIVALENT RELATIVE TO E1	1344
0509	10	0294	0167		C A	10	XXXX2	XXXX1		MAKE STORING ORDER)	1358
0629	21	0294	0947	0294		21	XXXX2		XXXX2	ALSO STORE FIRST LETTER OF ADDRESS)	1444
0589	10	0294	0399			10	XXXX2			FIRST CHARACTER INTO UPPER	1470
1263	20	0294	1247	0294		20	XXXX2		XXXX2	STORE THE COUNT	1710
1341	10	0294	0649			10	XXXX2			WHAT CELL DID WE RESERVE)	1754
1231	20	0294	1297	0294		20	XXXX2		XXXX2	STORE DYNAMIC LEVEL, RIGHT END WORD	1994
1086	15	0294	1049		J	15	XXXX2	B		PERHAPS MODIFY DYNAMIC LEVEL)	2150
0145	69	0298	0400		N 899	69		SUBR5		PROCESS D BACKWAROS	772
0750	21	0304	0458	0304	R XX	21	UH		UH	REPLACE DIGIT OF AVAIL. WORD BY W)	998
0337	10	0304	0359			10	UH	SPR		REPLACE DIGIT OF AVAIL. WORD BY W)	1008
0603	69	0306	0045		N SHX	69		SUB20		PROCESS D FORWARDS	766
0657	69	0310	0042		C NO	69		SUB19		STARTED WITH L, OR NOT FIRST. PROC. L	582
0563	11	0316	0331			11	C8	SU		LAST LINE AND END OF WORD. MODIFY)	1122
0214	69	0317	0142		J	69		SUBR6		PROCESS I BACKWARDS AGAIN	864
0115	45	0318	0026			45		QUITT		IFSO, QUIT SEARCH	482
0217	14	0320	1036			14	501XX			SAVE DYNAMIC LEVEL OF L FROM 0)	820
0267	14	0320	1080			14	501XX			WHICH L IS LESS, MEASURED ON CIRCLE)	836
0649	19	0320	1074			19	501XX			WHAT CELL DID WE RESERVE)	1756
1017	15	0320	1325			15	501XX			ADD 50 TO MAKE SURELY POSITIVE	1904
1085	14	0320	1272		C CP	14	501XX			NEW LEVEL, MODULO 50 IN UPPER	1914
1264	10	0320	0730			10	501XX			ADVANCE IT	2390
0569	10	0322	8003			10	QSB	8003		MAKE DRUM AVAILABLE)	978
0671	46	0324	0225			46		TP		D, NOT LAST LINE.	1022
0272	65	0325	0611		R M0005	65		SUB11		NEW SYMBOL. FIND BEST EQUIVALENT	1194
0723	10	0326	0281			10	W			REPLACE DIGIT OF AVAIL. WORD BY W)	1004
0533	24	0326	0479	0326	C RU	24	W		W	STORE 0 OR 1	1038
0653	90	0329	0521			90	X1	X0004		FIRST, UNFIXED. OTHER, FIXED.	702
0379	45	0332	0483			45		OVR		BRING BACK A SET OF RESULTS)	916
0681	11	0334	0539			11	9060			SUBTRACT 9060	1398
0504	90	0335	0459			90	SB10C			O, WE SHOULD STORE THIS DYNAMIC LEVEL	1274
0383	96	0336	0188			96		B		D, WITH I. I, WITH L.	610
0034	99	0338	0239		N XX8	99		9X8		MULTIPLE BRANCH ACCORDING TO TAGS)	744
0637	65	0340	8001			65		8001		BRINGING ORDER INTO LOWER	1514
0338	98	0341	0393			98	888	898		MULTIPLE BRANCH ACCORDING TO TAGS)	746
0743	65	0346	0551		SW	65	AI	SWOF		NOT PERFORMED---JUST FOR OPTIMIZING	1646
0737	65	0346	0551		J OFF	65	AI	SWOF		HAVE WE REACHED TOP OF TABLE	1648
1020	20	0346	8001	0346	C SAI	20	AI	8001	AI	STORE BRINGING ORDER, AND DO IT	1656
0739	65	0346	0601		J OFB	65	AI			START BACKWARDS SEARCH. HOW FAR	1670
0516	65	0346	0651		J ON	65	AI			AFTER RESTART. BRING BRINGING ORDER	1684
1225	15	0346	0751			15	AI			NEW AVAILABILITY WORD INTO DRUM)	1732
0755	65	0346	0951		C SKP	65	AI			ALL CASES. WHAT DID WE RESERVE)	1740
0293	44	0347	0248			44		OUT		BRING BACK A SET)	654
0245	46	0348	0199			46	AEB			WHICH L IS LESS, MEASURED ON CIRCLE)	842
0449	46	0352	0791			46		FULL		I, TABLE IS FULL	1526
0198	20	0353	0106	0353	N OUT	20	VAR		VAR	STORE VARIABLE ORDER	566
0206	65	0353	0229		J	65	VAR	LOOP		BRING VARIABLE ORDER, AND REPEAT	594
0252	11	0355	0259			11	O2			STORE PROCESSED CARO)	884
0753	69	0356	0305		J	69		SUBR4		PROCESS L BACKWARDS FROM D	814
0511	22	0359	0262	0359		22	SPR		SPR	MAKE SEVERAL VARIABLE ORDERS)	1090
0225	65	0359	0413		N TP	65	SPR			BOTTOM LINE OF TABLE	1094
0404	20	0359	0312	0359		20	SPR		SPR	MODIFY TO TAKE NEXT COLUMN OF TABLE)	1106
0561	24	0359	0362	0359		24	SPR		SPR	LAST LINE AND END OF WORD. MODIFY)	1118
0757	69	0360	0013			69	BRNG			BRING BACK A SET)	636
1061	91	0364	0688			91		F0		O, IT IS A BRANCH OPERATION	1812
0112	11	0365	8003			11	QSA	8003		CLEAR REGION AND SYMBOL TABLES)	960

0663 46 0366 0467		46	RC		0, ORUM. I, CORE.	1386
0775 15 0378 8002		15		8002	MAKE BRINGING ORDER	1566
0429 10 0382 8003		10		8003	MAKE ORUM AVAILABLE)	970
0731 15 0384 8002		15 T		8002	ADVANCE LOWER AND BRING ANOTHER	1432
0741 24 0384 0587	0384	24 T			PRESET T AS POSITIVE 1)	1504
0643 20 0384 0637	0384	20 T		T	SHORT, PRESET T AS -1)	1512
0433 96 0386 0288		96	L		0, WITH I. I, WITH L.	670
0388 69 0391 0244		69 ORCEB			SAVE IN CASE I IS CORE)	802
0600 24 0391 8002	0391	24 ORCEB		8002	SPEC. SUBR. BRING BACK BLANK + ORCEB	874
0342 20 0391 0335	0391	20 ORCEB		SB10C	STORE DYNAMIC LEVEL OF CORE ADDRESS	1206
0374 69 0391 0371		69 ORCEB		J0004	CORE. BRING DYNAMIC LEVEL	1320
	R J0007					
0439 15 0392 0747		15 SS			SYMBOL-STORING ORDER)	1332
0591 10 0394 0509		10 P3	A		MAKE STORING ORDER	1362
0642 16 0395 0449		16 ZTBL				1524
0194 69 0397 0400	J	69 LNM		SUBR5	PROCESS 0 BACKWARDS	760
0306 69 0397 0142	J	69 LNM		SUBR6	PROCESS 1 BACKWARDS	768
0101 65 0397 0201	J	65 LNM		SUB22	SAVE ORCEQ	776
0749 92 0402 1105		92 SHOP			0, IT IS A SHIFT OPERATION	1800
1051 92 0402 0656		92 SHOP			0, IT IS A SHIFT OPERATION	1808
0302 10 0405 0309		10 QA			CLEAR REGION AND SYMBOL TABLES)	956
0953 69 0406 0305	J	69		SUBR4	PROCESS L BACKWARDS FROM I	832
0506 11 0409 0313		11 20001			IS INITIAL ADDRESS LESS THAN 2000	1044
0777 11 0409 0663		11 20001			SUBTRACT 2000	1384
0676 10 0409 0713		10 20001	RES		RESTORE RELOCATED ORUM ADDRESS	1392
0964 16 0409 1064		16 20001			CORE OR ORUM	2364
1082 15 0409 0124	C CO	15 20001	Q0007		RESTORE THE 2000 AND JUMP BACK	2372
0907 16 0410 0315		16 QF			IS THIS LAST SET)	664
0558 46 0411 0212		46 OK			I, LAST ONE WOULD BE OVER 1999	1050
0559 46 0412 0613		46	FXT		I, FIRST IS NON-BLANK. FIXED ADDRESS.	1378
0913 46 0416 0567		46 SYM		L00	D, ADDRESS IS SYMBOLIC.	1450
0485 44 0416 0240		44 SYM			ARE LAST FOUR CHARACTERS DIGITS)	1458
0724 46 0416 0328		46 SYM			0, FIRST IS SPEC. CHAR. SYMBOLIC	1480
0923 69 0426 0441	R C0006	69		SUB13	800X ADDRESS. GET DYNAMIC LEVEL	1252
0324 10 0427 0331		10 800	SU		RESTORE, AND TAKE WORD FROM NEXT LINE	1024
1275 15 0428 8002		15		8002	MAKE BRINGING ORDER	1840
0432 65 0435 0439		65 LSymb			SYMBOL-STORING ORDER)	1330
0442 65 0435 0489		65 LSymb			LOCATION OF EQUIVALENT RELATIVE TO E1	1338
0345 20 0435 0588	0435	20 LSymb		LSymb	STORE ADDR OF SPACE RELATIVE TO START	1520
0754 20 0435 0638	0435	20 LSymb		LSymb	ADDRESS OF SYMBOL RELATIVE TO START	1534
1291 15 0444 8002		15		8002	MAKE SHIFTING ORDER	1726
0244 24 0447 0450	0447	24 T2		T2	SAVE IN CASE I IS CORE)	804
0282 69 0447 0600		69 T2			SPEC. SUBR. BRING BACK BLANK + ORCEB	872
0957 69 0460 0063		69		SUBR9	ANALYZE I, FOR TYPE OF ADDRESS	692
1310 44 0464 0514		44	UM		0, 0 TAG IS NON-ZERO	1954
0963 46 0466 0416		46	SYM		I, FIRST CHAR IS DIGIT. ADDR SYMBOLIC	1476
0315 45 0468 0319		45 NO			0, NO. I, YES.	666
0931 96 0484 0736		96 OI			I, ADDRESS IS BACKWARD I OR FORWARD L	1614
0459 20 0486 0335	0486	20 SAVOR	SB10C	SAVOR	STORE DYN LEV OF CORE, AND FINISH UP	1276
0645 24 0486 8002	0486	24 SAVOR	8002	SAVOR	STORE ORCEQ FOR USE WHEN WE START FORM	2242
0483 69 0486 0339		69 SAVOR			FINISHED. SAVE SAVOR AS ORCEQ)	936
0285 11 0488 0493	X	11 Q			BRING BACK A SET OF RESULTS)	928
0787 97 0490 0742	N 9	97 FIBO	BL		0, FORWARD I. I, BACKWARD L.	1784
0489 14 0492 1029		14 20XXX			LOCATION OF EQUIVALENT RELATIVE TO E1	1340
0638 14 0492 0904		14 2DXXX			DIVIDE BY 2	1536
0443 44 0497 0398		44	OUT2		STORE PROCESSED CARD)	888
0445 16 0498 0754		16 JBRL			SUBTRACT INITIAL BRINGING ORDER	1532
0099 69 0502 0740	R 0099	69	SUB		RBR ENTRY. TO SUBROUTINE	2454

0303 90 0507 0258		90	SKP	D, ONLY ONE.	534
1057 69 0510 0063		69	SUBR9	ANALYZE 0	716
1009 45 0512 1113		45	MAX	I, WE HAVE REACHEO TOP OF TABLE	1652
0162 16 0515 0619		16 110TH		REDUCE NUMBER YET TO BE DONE)	1014
0431 69 0515 0533		69 110TH	RU	BRING UNITY TO INOICATE UNRESERVATION	1036
0669 16 0515 0719		16 110TH		MODIFY TO ENO WITH 1999)	1054
0537 61 0515 1069		61 110TH	00N	MODIFY EXIT AND PREPARE TO QUIT	1438
0691 16 0515 1269		16 110TH		GET EQUIVALENT OF REGIONAL ADDRESS)	1492
0352 60 0515 1069		60 110TH	D0N	MODIFY EXIT, AND PREPARE TO QUIT	1528
1077 15 0515 8002	C PAKT	15 110TH	8002	MODIFY EXIT)	1706
1260 65 0515 1211		65 110TH	BT	COUNT OF I TAGS IN LOWER	1950
0464 15 0515 0514		15 110TH	UM	COUNT OF TAGS IN LOWER	1956
1042 65 0515 0178		65 110TH	EXITY	UNIT CORRECTION TO DYNAMIC LEVEL	2010
0698 66 0515 0178		66 110TH	EXITY	IF 8001, CORRECTION IS -1	2018
1113 69 0516 1070	N MAX	69 ON		AT TOP. SET SWITCH ON)	1678
1213 69 0516 1120	N MIN	69 ON		SECONO PART, BACKWARDS. CHANGE SWITCH	1692
0763 44 0517 0618		44 NZ		0, IT WAS NOT RIGHT ONE	1424
0618 46 0517 0972		46 NZ	DEF	0, IT WAS NOT RIGHT ONE . I, IT AS.	1426
0530 91 0526 0525	R X0013	91 X0009	X0008	CORE. BRANCH, LIKE BLANK. ARITH, UNF	732
0981 97 0534 0786		97	BAK	D, FORWARD. I, BACKWARD.	1632
0633 65 0536 0220	J	65 EQUIV	F0003	BRING EQUIV AND MULTIBRANCH	1152
0908 65 0536 0271	J PRE	65 EQUIV	M0004	EQUIVALENT TO LOWER. MULTIBRANCH	1178
0711 65 0536 0921	J PRE	65 EQUIV	C0004	BRING EQUIVALENT. MULTIBRANCH	1228
0911 65 0536 0371	J PRE	65 EQUIV	J0004	BRING EQUIVALENT. MULTIBRANCH	1296
0674 65 0536 0541		65 EQUIV		BRING EQUIVALENT	1352
1073 65 0536 0591	N IPOS	65 EQUIV		USE I-POSITION, BRING EQUIVALENT	1360
0641 20 0536 0589	0536	20 EQUIV	EQUIV	STORE THESE 4 OIGITS	1468
0668 15 0536 0691		15 EQUIV		GET EQUIVALENT OF REGIONAL ADDRESS)	1490
1269 23 0536 0704	0536	23 EQUIV	ABS	STORE EQUIVALENT	1494
0776 23 0536 0704	0536	23 EQUIV	ABS	STORE EQUIVALENT OF SYMBOL	1554
1031 20 0536 0178	0536	20 EQUIV	EXITY	STORE THIS ADDRESS IN EQUIV	1760
1033 16 0536 0992	C EO	16 EQUIV		BOTH. SUBTRACT THE 800X ADDRESS	2004
1347 16 0536 1092		16 EQUIV		SUBTRACT THE ADDRESS IN QUESTION	2014
1015 65 0536 0421	J PRE	65 EQUIV	L0004	BRING EQUIVALENT. MULTIBRANCH	2074
1065 65 0536 0472	J PRE	65 EQUIV	00004	BRING EQUIVALENT. MULTIBRANCH	2122
1215 65 0536 0575	J	65 EQUIV	10004	BRING EQUIVALENT. MULTIBRANCH	2184
1217 65 0536 0120	J	65 EQUIV	00003	BRING EQUIVALENT. MULTIBRANCH.	2318
0127 20 0536 1289	0536	20 EQUIV	SD	OLD SYMBOL. STORE I AS NEW EQUIV	2344
0128 20 0536 1289	0536	20 EQUIV	SD	NEW SYMBOL. STORE I AS ITS EQUIVALENT	2346
0129 20 0536 1289	0536	20 EQUIV	SD	800X. STORE ITS EQUIVALENT.	2348
0130 20 0536 1289	0536	20 EQUIV	SD	CORE. STORE I AS ITS EQUIVALENT.	2350
0131 20 0536 1289	0536	20 EQUIV	SD	OTHER. STORE I AS THE EQUIVALENT.	2352
0385 69 0538 0441		69	SUB13	FINO DYNAMIC LEVEL OF CORE ADORESS	1202
1037 14 0540 1048		14 42N0		SHOULO WE INTERCHANGE ADOENOS)	1872
0539 46 0542 0977		46	BA0	I, RELOCATED CORE ADDRESS TOO HIGH	1400
0942 16 0545 1085		16 COUNT	CP	IF BACKWARD L, SUBTRACT TAG-COUNT)	1912
0514 20 0545 0598	0545	20 COUNT		STORE TAG-COUNT	1958
1337 15 0545 0335	C UM C MW	15 COUNT	SB10C	CORE OR DRUM. AOO TAG-COUNT TO DYN LV	2108
1092 45 0546 0698		45 NZ		0, ADDRESS IS NOT 8001	2016
0296 15 0549 0954		15 601		INITIAL OF VARIABLE BRINGING OROER	1638
0248 20 0553 0256	0553	20 VAR	VAR	STORE VARIABLE OROER	658
0009 65 0553 0907	J	65 VAR		IS THIS LAST SET)	662
0570 65 0553 0243		65 VAR	LOOP	BRINGING OROER. BACK FOR ANOTHER CARD	894
0451 10 0554 0509		10 P2	A	MAKE STORING OROER)	1356
0273 01 0555 0273	R M0006	01 0555	M0006	800X ADDRESS. STOP THE MACHINE	1198
0275 01 0555 0275	R M0008	01 0555	M0008	OTHER ADDRESS. STOP MACHINE	1208
1103 65 0556 0611		65	SUB11	FIND BEST EQUIVALENT TO BLANK	1188
1107 46 0560 0311	C BOT	46	2N0	0, WE WILL USE ONE MEASUREO FROM 0	848
1311 15 0564 8002		15	8002	BRINGING OROER	2042
0262 10 0565 0331		10 C3	SU	MAKE SEVERAL VARIABLE ORDERS)	1092
0615 69 0568 0677		69	IN0EX	TO SUBROUTINE 18 TO INDEX	1310
1081 60 0584 1089	N YES	60 9	AX	MORE TAGS INTO UPPER	1978
0781 44 0585 0686		44	SHRT	0, LONG SYMBOL	1500
1237 10 0590 8003		10	8003	MAKE BRINGING OROER	1962

0639 10 0592 8003		10	8003	MAKE BRINGING DRORER)	1484
1289 69 0594 0661	C SD	69	SUBR7	STORE SYMBOL AND EDUIVALENT IN TABLE	2354
1292 69 0595 1025		69	SUB2R	RESERVE ADDRESS	2090
1345 45 0596 0646		45	SDD	IF D IS BLANK, WE SHDULD STDR ZERD	2622
0493 44 0597 0448		44	DUT	BRING BACK A SET OF RESULTS)	930
0396 15 0599 1104		15 602		INITIAL BRINGING DRDR)	1662
0300 95 0603 0255		95 SHX		D, ND, I, YES.	688
0204 11 0607 0161	X	11 D		BRING BACK A SET)	558
1307 11 0610 0415		11 DB		MAKE DRUM AVAILABLE)	974
1059 45 0612 1213		45	MIN	D, WE ARE NDT BACK TO START YET	1674
0269 60 0622 0627	R M0002	60 87THX	SB10B	N. G. DMIT PUNCHING	1182
0977 60 0622 0177	N BAD	60 87THX	EXITX	MODIFIED ADDR TOO HIGH. 8000 TD UPPER	1412
0419 60 0622 0627	R L0002	60 87THX	SB10B	N.G. FIX TD OMIT PUNCHING	2078
0921 69 0624 0677	N C0004	69	INDEX	DRUM DR EOUVALENT. INDEX IF TAGGED	1244
0922 65 0625 0611	R C0005	65	SUB11	NEW SYMBOL. FIND BEST EOUVALENT	1248
0772 90 0626 0369		90	J0002	D, DRUM IS NDT FULL	1304
0030 69 0633 0063		69	SUBR9	TD SURDRUTINE 9 TD ANALYZE THE ADDRESS	1150
0482 45 0636 0537		45 ABC		I, THE ADDRESS IS BLANK	1436
1287 45 0640 1042		45 XY		I, 8002-8003 + WRONG PARITY	2008
1339 45 0644 0694		45 ALL		I, IT IS ANDTHER RBD CARD	2406
0497 10 0650 8003		10 QS2	8003	STDR PROCESSED CARD)	890
0399 44 0654 0704		44	ABS	I, ZERD. ADDRESS WAS ABSOLUTE	1472
0701 45 0655 0705		45	FULL	I, THERE IS ND AVAILABLE LDCATIDN.	1688
0403 90 0657 0308		90 NO		D, IT IS NDT	576
1053 21 0658 0461	0658	21 X	X	MAKE SEVERAL VARIABLE DRDRS)	1064
0401 10 0658 0363		10 X		MAKE SEVERAL VARIABLE DRDRS)	1078
1105 93 0660 0710		93 XAS		D, IT IS AN INDEXING OPERATIDN	1802
0656 93 0660 1061		93 XAS		D, IT IS AN INDEXING OPERATIDN	1810
0531 01 0666 0531	R X0014	01 0666	X0014	DTHER. ERROR. SHDULD NDT BE HERE.	734
1312 69 0666 1024		69	SUBR8	PRDCESS D	2268
0665 45 0668 1219		45	NG	I, REGIDN IS UNOEFINED	1488
0919 60 0672 0627	R C0002	60 88THX	SB10B	N G. OMIT PUNCHING	1232
0470 60 0672 0627	R D0002	60 88THX	SB10B	N.G. DUIT AND DMIT PUNCHING	2126
0620 60 0673 0057	C LOOP	60	SB10A	TD SUB 10A TD PUNCH DNE CARD	910
0372 65 0675 0611	R JD005	65	SUB11	NEW SYMBDL. FIND BEST VALUE	1314
1123 46 0676 0977		46	BAD	I, RELDCATEO DRUM WILL EXCEED 1999	1390
0280 69 0683 0586		69	SUB17	CDRRECTIDN TD DYNAMIC LEVEL	1256
1216 46 0684 0029		46	0029	IF A DIGIT, TD BLR WITHDUT DESIGNATING	2480
0484 69 0687 0390	N DI	69 F		PREPARE EXIT ANO GD TO SUBROUTINE 13)	1618
0735 97 0688 0490		97 FD	FIBD	D, FDRWARD D. I, BACKWARD D.	1782
0364 99 0688 0490		99 FD	FIBD	I, BACKWARD L, BRANCH, UNFIXED D	1814
1334 15 0695 1299		15 C3		MODIFY IDENTIFICATIDN OF WORD	2562
0745 15 0695 1349		15 C3		MODIFY IDENTIFICATIDN OF WORD	2566
0795 15 0695 0652		15 C3		MODIFY IOENTIFICATIDN OF WORD	2570
0597 10 0700 0547		10 QS	A	BRING BACK A SET DF RESULTS)	932
0945 10 0702 8003		10 C6	8003	MODIFY BRINGING ORDER, AND BRING	2576
0995 10 0702 8003		10 C6	8003	MODIFY BRINGING DRDR, AND BRING	2580
1045 10 0702 8003		10 C6	8003	MODIFY BRINGING DRDR, AND BRING	2584
1095 10 0702 8003		10 C6	8003	MODIFY BRINGING DRDR, AND BRING	2588
0503 90 0707 0358		90	A	IF NDT, JUMP AHEAD	606
0354 69 0708 0511		69 C2		MAKE SEVERAL VARIABLE DRDRS)	1088
0413 16 0708 0463		16 C2		IS IT ALSO RIGHT END DF WRDR	1096
0958 69 0711 0063		69 PRE	SUBR9	ANALYZE THE ALPHABETIC D ADDRESS	1226

0926	69	0711	0661	N C0009	69	PRE	SUBR7	STORE EQUIVALENT OF NEW SYMBOL	1282
0715	15	0718	8002			15	8002	MAKE BRINGING ORDER	1540
0369	60	0722	0627	R J0002	60	89THX	SB10B	N.G. OMIT PUNCHING	1300
0573	60	0722	0627	R I0002	60	89THX	SB10B	N.G. QUIT AND OMIT PUNCHING	2108
1223	10	0726	0681			10	CDIFF	ADD RELOCATION AMOUNT	1396
0968	15	0726	1082			15	CDIFF	ADD CORE RELOCATION AMOUNT	2368
0946	20	0726	0001	0726 C SCD	20	CDIFF	SETCC	CDIFF STORE CORE RELOCATION AMOUNT	2634
0924	69	0727	0677	R C0007	69		INOEX	CORE ADDRESS. INOEX IT	1262
1224	90	0728	0419			90	L0002	O, DRUM NOT FULL. I, DRUM IS FULL.	2082
0929	46	0732	0933			46	DRM	I, CORE. D, DRUM.	2052
0779	20	0733	0936	0733		20	B	SHOULD WE INTERCHANGE ADDENDS)	1860
1010	15	0733	0987			15	B	SHOULD WE INTERCHANGE ADDENDS)	1868
0917	16	0733	0935			16	B	INTERCHANGE AND MAKE NEGATIVE)	1880
0532	95	0735	0787			95	9	D, D ADDRESS. I, FORWARD I OR BACKW L	1780
0534	69	0737	0440			69	OFF	FORWARD. SET SWITCH OFF FOR 1ST PART	1634
0786	69	0739	0692	N BAK	69	OFB		GOING BACKWARD. SET SWITCH OFF)	1658
0440	24	0743	0296	0743		24	SW	FORWARD. SET SWITCH OFF FOR 1ST PART	1636
0692	24	0743	0396	0743		24	SW	GOING BACKWARD. SET SWITCH OFF)	1660
1070	24	0743	0446	0743		24	SW	AT TOP. SET SWITCH ON)	1680
1120	24	0743	0496	0743		24	SW	SECOND PART, BACKWARDS. CHANGE SWITCH	1694
1046	44	0752	0003			44	REAOC	D, WE ARE NOT DONE.	2594
0500	69	0753	0400			69	SUBR5	PROCESS O BACKWARDS	812
1100	90	0755	0905			90	SKP	D, SKIP BECAUSE COMP IS DOUBLE	1714
0454	69	0758	0561			69	C7	LAST LINE AND END OF WORD. MODIFY)	1116
1058	65	0761	0611	N BB1	65		SUB11	GET BLANK O DYNAMICALLY FROM I	1240
0512	15	0765	1020			15	800	ADVANCE VARIABLE BRINGING ORDER	1654
1265	65	0768	0611	N DNB	65		SUB11	USUAL CASE, I BLANK AND NOT D.	2200
0216	15	0769	0973			15	C5	MODIFY TO TAKE NEXT COLUMN OF TABLE)	1100
1220	14	0774	1021			14	41XXX	WHAT CELL DID WE RESERVE)	1746
0474	69	0778	0441	R D0006	69		SUB13	800X ADDRESS. FIND DYNAMIC LEVEL	2144
0767	92	0785	1222			92	SEO	JUMP UNLESS OPERATION 80,82, OR 88	1846
0588	46	0791	0642			46	FULL	O, TABLE IS FULL	1522
0937	20	0792	0495	0792		20	INO	BACKWARD. SHOULD WE INTERCHANGE ADDENDS	1854
1351	66	0792	0935	N OK	66	INO	P	MAKE NEGATIVE, BUT DO NOT INTERCHANGE	1882
0690	69	0794	1025			69	SUB2R	RESERVE 0000	2430
0746	45	0796	0946			45	SCO	JUMP IF I IS BLANK	2630
0590	65	0800	1306	J	65	0800		BRING OPTIMIZING ADDENDS AND TAGS	1964
0748	94	0800	0912		94	0800		IF BLANK O AND MDF, STOP MACHINE	2130
0778	00	0808	0782	J	00	0808		TAGS AND EXIT FOR USE IN SUBR 11	2146
1243	00	0808	0798	J	00	0808		TAGS AND EXIT FOR SUBROUTINE 13	2160
1293	00	0809	0998	J	00	0809		TAGS AND EXIT FOR SUBROUTINE 13	2222
1115	00	0888	0469	J	00	0888	00001	TAGS AND EXIT FOR SUBROUTINE 11	2134
1276	00	0888	0477	J	00	0888	D0009	TAGS AND EXIT FOR USE IN SUBR 11	2142
0768	00	0889	0572	J	00	0889	I0001	TAGS AND EXIT FOR SUBROUTINE 11	2202
0979	00	0889	0580	J	00	0889	I0009	TAGS AND EXIT FOR SUBR 11.	2210
1342	00	0890	0418	J	00	0890	L0001	TAGS AND EXIT FOR USE IN SUBR 11	2100
0249	22	0901	0354	0901	22	0901		0901 MAKE SEVERAL VARIABLE ORDERS)	1086
0299	22	0901	0404	0901	22	0901		0901 MODIFY TO TAKE NEXT COLUMN OF TABLE)	1104
0349	24	0901	0454	0901	24	0901		0901 LAST LINE AND END OF WORD. MODIFY)	1114
0053	60	0902	0007		60	0902		IS O INOEXEO	386
0055	60	0902	0307		60	0902		IS O INOEXEO	464
1006	21	0902	1056	0902	21	0902		STORE O TAG	1940
1211	10	0902	1310		10	0902		O TAG INTO UPPER	1952
0564	65	0902	0662	C BT	65	0902		BRING APPROPRIATE TAG	2044
0318	65	0903	0457	J	65	0903		IS I INOEXEO	484
0628	20	0903	1106	0903	20	0903		0903 STORE I TAG	1946

0949 20 0906 1010	0906	20 A		A	SHOULD WE INTERCHANGE ADDENDOS)	1866
1052 66 0906 1111		66 A			INTERCHANGE AND MAKE NEGATIVE)	1876
0455 69 0908 0063		69 PRE	SUBR9		ANALYZE THE ADDRESS	1176
0505 69 0908 0661		69 PRE	SUBR7		USUAL CASE. STORE SYMBOL AND EQUIV	1214
0426 00 0908 0280	J	00 0908			TAGS AND EXIT FOR SUBROUTINE 13	1254
0190 00 0908 0344	J	00 0908			TAGS AND EXIT FOR SUBROUTINE 13	1270
0538 00 0909 0342	J	00 0909			TAGS AND EXIT FOR USE IN SUBR 13	1204
1108 69 0911 0063		69 PRE	SUBR9		ANALYZE ALPHABETIC I	1294
0368 69 0911 0661	N J0001	69 PRE	SUBR7		STORE EQUIVALENT OF NEW SYMBOL	1298
0612 15 0915 1020		15 599	SAI		RESTORE AND MODIFY	1676
0452 10 0916 0980		10 Q			DUPLICATE INTO AVAILABILITY TABLE)	2414
0475 69 0928 0677	R 00007	69	IN0EX		CORE ADDRESS. IN0EX IF TAGGED	2152
0766 16 0930 1339		16 94I			IS THIS ANOTHER RBO CARO)	2404
0582 97 0935 0937		97 P			O, FORWARD. I, BACKWARD.	1852
1240 69 0944 0042		69	SUB19		PROCESS L, IF REGIONAL OR SYMBOLIC	2532
0550 69 0953 0142		69	SUBR6		PROCESS I BACKWARDS	830
0751 69 0955 1209		69 STR			NEW AVAILABILITY WORD INTO ORUM)	1734
0954 20 0959 0462	0959	20 BP1		BP1	STORE VARIABLE BRINGING ORDER	1640
1104 20 0959 0562	0959	20 BP1		BP1	INITIAL BRINGING ORDER)	1664
0655 15 0959 1020		15 BP1	SAI		RESTORE AND MODIFY	1690
1072 90 0976 1077		90	PAKT		I, YES.	1610
1274 90 0978 0573		90	I0002		I, ORUM IS FULL	2196
0576 65 0979 0611	R I0005	65	SUB11		NEW SYMBOL. FIND BEST VALUE.	2208
1078 69 0982 0677		69	IN0EX		IN0EX 0	2272
0761 00 0988 0918	J	00 0988	C0001		TAGS AND EXIT FOR SUBROUTINE 11	1242
0625 00 0988 0926	J	00 0988	C0009		EXIT AND TAGS FOR SUBROUTINE 11	1250
0556 00 0989 0268		00 0989	M0D01		TAGS AND EXIT FOR SUBROUTINE 11	1190
0325 00 0989 0276	J	00 0989	M0009		TAGS AND EXIT FOR USE IN SUBROUTINE 11	1196
0675 00 0990 0368	J	00 0990	J0001		TAGS AND EXIT FOR SUBROUTINE 11	1316
0084 65 0990 8002	R 0084	65	8002		ADDING ORDER TO LOWER	2498
0797 15 1000 0555		15 P1			MAKE BRINGING ORDER)	1346
0983 90 1001 0738		90 M0FLL			O, START SEARCH. I, WE ARE DN QUITT	2096
09D0 65 1003 0508	R 0900	65 N			HAVE WE FINISHED	1010
0619 20 1003 0456	1003	20 N		N	REDUCE NUMBER YET TO BE DONE)	1016
0387 20 1003 0506	1003	20 N		N	STORE N-1	1042
0166 10 1003 0558		10 N			UPPER, LAST ADDRESS - 2000	1048
0719 20 1003 0411	1D03	20 N	DK	N	MODIFY TO END WITH 1999)	1056
1D01 69 1004 0107	N M0FLL	69 L	A		STARTED WITH NEW-SYMBOL L	402
1002 44 10D5 0606	C LDDP	44	UNO		I, SYMBOL IS NOT IN THE TABLE	1420
1353 90 1008 1058		90	BB1		D, THERE WAS A BLANK BACKWARDS L	1236
1060 44 1014 0414		44	EVN		O, DPREG WAS 000. I, EVEN.	1890
0712 69 1015 0063		69 PRE	SUBR9		ANALYZE L-ADDRESS	2072
0418 69 1015 0661	N L0001	69 PRE	SUBR7		SUB 11 FOUND EQU OF SYMB. RESERVE	2076
1114 69 1018 1025		69	SUB2R		RESERVE FIRST CELL IN BAND	2382
0366 10 1019 1123		10 D0IFF			ORUM. ADD RELOCATION AMOUNT	1388
1317 15 1019 1082	N D	15 D0IFF	C0		ADD ORUM RELOCATION AMOUNT	2370
0502 10 1019 1232	J	10 D0IFF	ZQ		ADD RELOCATION AMOUNT	2456
0646 20 1019 0696	1019 C 500	20 D0IFF		DDIFF	STORE ORUM RELOCATION AMOUNT	2626
1219 61 1022 1069		61 21XXX	DDN		MODIFY EXIT FOR N G SYMBOL	1558
0635 15 1022 1027	C NG	15 21XXX			ADVANCE CARO NUMBER)	1590
0967 14 1022 1060		14 21XXX			WHICH ONE OF WE USE)	1888
1297 14 1022 1322		14 21XXX			DIVIDE DYNAMIC LEVEL BY 2	1996
0918 20 1023 0376	1023 N C0D01	20 BLANK	A		BLANK STORE EQUIV OF BLANK FOR FORWARD L	1230
0728 65 1023 0421		65 BLANK	L0D04		USE PROPER VALUE FOR BLANK ADDRESS	2084
0469 20 1023 0476	1023 N D0001	20 BLANK	00008		BLANK STORE EQUIV OF BLANK FOR REFERENCE	2124
0572 20 1023 0577	1023 N I0001	20 BLANK	10006		BLANK STORE EQUIVALENT OF BLANK FOR REFERENCE	2186
0978 65 1023 0579		65 BLANK	10008		MAKE BLANK I EQUAL TO BLANK 0	2198
0575 69 1028 0677	N I0004	69	IN0EX		ORUM ADDRESS. IN0EX IF TAGGED	2204
0422 69 1030 0983	R L00D5	69 1030			NEW SYMBOL. WHERE ARE WE	2094

0984	24	1030	1084	1030		24	1030		1030	TAG IN CASE L IS UNDEFINED SYMBOL)	2524
0047	24	1030	0033	1030		24	1030		1030	DELETE OUITT TAG)	374
0107	24	1030	0083	1030	C A	24	1030		1030	STORE TAG TELLING HOW SEARCH STARTED	404
0207	69	1030	0133			69	1030			IF SD, WHERE DID SEARCH START	428
0507	69	1030	0283			69	1030			WHERE DID SEARCH START	536
0308	69	1030	0333			69	1030			IF SD, WHERE DID WE START	578
0707	69	1030	0383			69	1030			IF SD, WHERE DID SEARCH START	608
0319	69	1030	0433			69	1030			IF SD, WHERE DID SEARCH START	668
1278	69	1032	0677			69		INDEX	INDEX I		2292
0640	65	1043	1347		N XY	65	8001			8001 INTO RIGHT END LOWER	2012
1041	69	1044	1047			69	81STX			RESTORE FIRST CARD TAG)	366
0008	69	1044	0047		N 0008	69	81STX			DELETE OUITT TAG)	372
0139	69	1044	0297		J	69	81STX			CHANGE TAG TO SECOND-CARD TAG)	588
0234	69	1044	0647			69	81STX			PRESET DRUM TAG)	964
0270	69	1044	0697		R M0003	69	81STX			BLANK, CHANGE FIRST-CARD TAG)	1184
0709	15	1044	0499		C SU810	15	81STX			C.1. TO PUNCH X-9)	1582
0021	69	1044	1298		R 0021	69	81STX			BLA ENTRY, CHANGE DRUM TAG	2446
1047	24	1050	0160	1050		24	1050	R0010		1050 RESTORE FIRST CARD TAG) MULTIBRANCH	368
0056	69	1050	0203			69	1050			IS THIS FIRST SET OF SEARCH)	424
0017	24	1050	0253	1050		24	1050			1050 CHANGE FIRST-CARD INDICATOR TO 2ND)	490
0026	69	1050	0303		N OUITT	69	1050			WAS THERE ONLY ONE SEARCHED CARD	532
0279	69	1050	0403		J	69	1050			IS THIS FIRST OUITT SET	574
0297	24	1050	0453	1050		24	1050			1050 CHANGE TAG TO SECOND-CARD TAG)	590
0019	69	1050	0503		N BACKW	69	1050			WAS THERE JUST ONE SEARCH CARD	604
0067	24	1050	0358	1050		24	1050	A		1050 START ON L, FIRST-CARD TAG)	616
0519	69	1050	0653		R X0002	69	1050			BLANK ADDRESS, IS THIS FIRST CARD	700
0526	69	1050	0703		R X0009	69	1050			BLANK, IS THIS FIRST CARD	724
0697	24	1050	1103	1050		24	1050			1050 CHANGE FIRST-CARD TAG)	1186
0920	69	1050	1353		R C0003	69	1050			BLANK ADDRESS, WAS THERE BLANK BACK L	1234
0344	69	1050	0504			69	1050			FIRST-CARD TAG	1272
0201	69	1050	1062		N SU822	69	1050			FIRST-CARD TAG	2236
0551	16	1054	1009		SWOF	16	799			SUBTRACT END OF TABLE	1650
0496	65	1054	1020			65	799	SAI		START BACK FROM TOP OF TABLE	1696
0659	44	1063	0314			44		LH		O, ADDR IS ODD, EQUIV IN RIGHT END	1544
0762	69	1065	0063			69	PRE	SU8R9		ANALYZE D-ADDRESS	2120
0477	69	1065	0661		N D0009	69	PRE	SU8R7		STORE EQUIVALENT OF NEW SYMBOL	2170
1214	45	1068	1329			45		OUT		O, NOT BLANK, I, IT IS BLANK,	2386
0670	44	1073	0674			44	IP05			O, USE I-POSITION, I, USE D-POSITION,	1350
1322	44	1076	1026			44		EVN		O, DYNAMIC LEVEL IS ODD,	1998
0678	44	1081	0682			44	YES			I, D IS NOT TRUE ADDRESS	1974
0632	95	1085	1087			95	CP			IF BACKWARD L, SUBTRACT TAG-COUNT)	1908
1087	97	1085	0942			97	CP			IF BACKWARD L, SUBTRACT TAG-COUNT)	1910
0782	69	1086	0586			69		SU817		PERHAPS MODIFY DYNAMIC LEVEL)	2148
0934	15	1090	8002			15	K	8002		MAKE STORING ORDER, AND STORE ZERO	2510
1036	21	1091	1094	1091		21	OL0		OL0	SAVE DYNAMIC LEVEL OF L FROM O)	822
1038	16	1091	0245			16	OL0			WHICH L IS LESS, MEASURED ON CIRCLE)	840
0793	20	1097	1020	1097	C ST	20	AO	SAI	AO	STORE COMPARISON CONSTANT	1668
0651	16	1097	0701			16	AO			COMPARISON CONSTANT, SUBTRACTED	1686
0548	93	1101	0785			93		SEO		O, OPERATION 31, I, 30,35,36,	1828
0002	69	1112	0039		R 0002	69		SUB14		DP, C.1., TAGS, OPTIM, ALDPT	2248
0912	65	1115	0611			65		SUB11		BLANK O, BUT NOT MOD, FIND VALUE OF O	2132
0714	69	1117	1024			69		SUBR8		PROCESS I	2288
0466	10	1119	0724			10	291XX			ADD 29	1478
0684	15	1119	0734			15	291XX			ADD 29	2482
0410	69	1124	8003		O DF	69	S0000	8003			752
1271	11	1124	0929			11	27TH			SUBTRACT 2000, TO DETERMINE RANGE	2050
0086	24	1125	0028	1125	J SEND	24	S0001	X	S0001	STORE SET OF DATA)	412
0260	69	1125	8003		J BRNG	69	S0001	8003		BRING BACK A SET)	554
0211	24	1199	0252	1199	J SEN	24	SMAXM		SMAXM	STORE PROCESSED CARD)	882
0174	69	1199	8003		O DL	69	SMAXM	8003			944
0210	24	1200	0028	1200	D MAX	24	SMAX1	X	SMAX1	CONSTANT FOR WHETHER STORAGE IS FULL	518
0729	84	1200	0775			84	1200			SEARCH TABLE ACCORDING TO SIZE OF ADDR	1564
0604	16	1208	0913			16	909			SUBTRACT 9090908995	1448
0962	69	1215	0063			69		SUBR9		ANALYZE I-ADDRESS	2182

0764 69 1217 0063	C BOTH	69	SUBR9	ANALYZE I	2316
0609 16 1218 8002	R U0002	16 KA	8002	RESTORE LOWER, AND ADD NEXT SYMBOL	2512
0472 69 1226 0677	N 00004	69	IN0EX	ORUM ADDRESS. IN0EX IF TAGGED	2136
1117 44 1228 1278	J	44 NGI		O, RELOCATED I IS EXCESSIVE	2290
1026 65 1229 1033	N EVN	65 8003	EO	EVEN. PUT 8003 AT RIGHT END OF LOWER	2002
0966 16 1230 8002		16	8002	DUPLICATE INTO AVAILABILITY TABLE)	2422
0029 69 1232 0740	R 0029	69 ZO	SUB	BLR ENTRY. TO SUBROUTINE	2442
0010 69 1234 1244	J 0010	69 85TH		C.I. FOR AVAILABILITY TABLE)	2552
0682 60 1235 1089		60 8	AX	MORE TAGS INTO UPPER	1976
1088 44 1241 0743	R TA	44 YES	SW	IS A LOCATION AVAILABLE IN THIS GROUP	1644
0788 69 1243 0441		69	SUB13	FIND DYNAMIC LEVEL	2158
0752 10 1245 0952		10 OS		RESTORE VARIABLE BRINGING ORDER	2596
0644 61 1248 0966	N ALL	61 SEND		DUPLICATE INTO AVAILABILITY TABLE)	2420
1277 65 1250 0706	J	65 1250		BRING SHIFT ADDENDS	1824
1106 45 1260 1211		45	BT	O, I TAG IS NON-ZERO	1948
1012 44 1265 0616		44 DNB		O, O IS NOT BLANK.	2192
0552 15 1266 0784		15 ST		MAKE STORING ORDER	2488
0914 69 1267 0063		69	SUBR9	ANALYZE O	2336
0720 69 1273 1025		69	SUB2R	RESERVE IF ORUM ADDRESS	1406
0473 65 1276 0611	R 00005	65	SUB11	NEW SYMBOL. FIND EQUIVALENT	2140
0974 10 1277 8003		10	8003	MAKE BRINGING ORDER	1822
1076 65 1279 1033		65 8002	EO	000. PUT 8002 AT RIGHT END OF LOWER	2000
0980 46 1283 0634		46 DONE		DUPLICATE INTO AVAILABILITY TABLE)	2416
0938 69 1293 0441		69	SUB13	FIND DYNAMIC LEVEL	2220
0952 16 1295 1249		16 C7	LOOP	MODIFY IDENTIFICATION FOR NEXT CARD	2598
0910 84 1300 1275		84 1300		LOOK UP IN TABLE	1838
0790 16 1308 1216		16 99THX		SUBTRACT 90	2478
0654 11 1308 0963		11 99THX		NON-ZERO. SUBTRACT 90	1474
1262 69 1315 1024		69	SUBR8	PROCESS L	2254
0608 44 1316 0001	R U0001	44	SETCC	I, IT IS ZERO AND WE HAVE FINISHED	2502
1064 46 1317 0968		46 O		O, ORUM. I, CORE.	2366
1315 44 1321 1324	J	44 NGL		O, RELOCATED L IS EXCESSIVE	2256
0666 44 1326 1078	J	44 NGO		O, RELOCATED O IS EXCESSIVE	2270
0118 65 1328 1233	R 00001	65 83RD	BP	INVALID ADDRESS	2320
0119 65 1328 1233	R 00002	65 83RD	BP	BLANK I. CARD IS UNUSABLE	2322
0121 65 1328 1233	R 00004	65 83RD	BP	NEW SYMBOL I. CARD IS UNUSABLE	2326
0125 65 1328 1233	R 00008	65 83RD	BP	N G O. CARD IS UNUSABLE	2340
0126 65 1328 1233	R 00009	65 83RD	BP	BLANK O. CARD IS UNUSABLE	2342
0780 46 1329 0094		46 OUT	0094	O, WE HAVE FINISHED	2398
1016 45 1330 1332		45	BL	I, I IS UNPUNCHED	2462
0578 69 1331 0677	R I0007	69	IN0EX	CORE ADDRESS. IN0EX IF TAGGED	2214
1280 69 1333 0740		69	SUB	TO SUBROUTINE	2450
0738 65 1342 0611		65	SUB11	FIND AN EQUIVALENT	2098
1284 15 1344 1249		15 RS1	LOOP	INITIAL WORD SHOWING LOCATION OF TABLE	2558
1294 60 1348 1284		60 A1		VARIABLE BRINGING ORDER INTO UPPER	2556
1048 44 1351 1052		44 OK		I, WE SHOULD INTERCHANGE	1874
1096 11 1352 1046		11 O		ARE WE DONE	2592
0601 16 1354 1059		16 600		SUBTRACT COMPARISON CONSTANT	1672
0446 65 1354 1020		65 600	SAI	RESET BRINGING ORDER, AND SEARCH MORE	1682

0951 16 1354 1309					16 600		WHAT CELL OLD WE RESERVE)	1742
0915 60 1354 1088		0 599			60 A0000	TA		1762
0322 20 1355 0000	1355	0 05B			20 A0001		0000 A0001 RESTORE COMPARISON CONSTANT AND MODIFY	986
0565 20 1355 0900	1355	P C3			20 A0001		0900 A0001 INITIAL OF STORING ORDER	1128
1354 60 1355 1088		0 600			60 A0001	TA		1764
1230 69 1355 8003		J			69 A0001		8003 DUPLICATE INTO AVAILABILITY TABLE)	2424
1348 69 1355 0168		P A1			69 A0001	T0001	INITIAL VARIABLE BRINGING ORDER	2600
0610 20 1356 0000	1356	0 0B			20 A0002		0000 A0002 COMPARISON CONSTANT	984
0549 60 1356 1088		P 601			60 A0002	TA		1766
0599 60 1357 1088		P 602			60 A0003	TA		1768
1248 24 1359 1034	1359	J SEND			24 A0005	A	A0005 DUPLICATE INTO AVAILABILITY TABLE)	2426
0116 20 1551 0900	1551	0 796			20 A0197		0900 A0197 COMPARISON CONSTANT FOR LAST LINE	1132
0382 20 1554 1307	1554	J			20 A0200		A0200 MAKE ORUM AVAILABLE)	972
1054 60 1554 1088		0 799			60 A0200	TA		1770
0427 20 1555 0900	1555	Q 800			20 A0201		0900 A0201 TO RESTORE FOR NEXT LINE OF TABLE	1134
1000 69 1555 8003		P P1			69 E0001	8003	INITIAL OF BRINGING ORDER	1364
0554 22 1555 0179	1555	P P2			22 E0001	EXITZ	E0001 INITIAL OF ORDER TO STORE 0-POSITION	1366
0394 23 1555 0179	1555	P P3			23 E0001	EXITZ	E0001 INITIAL OF ORDER TO STORE 1-POSITION	1368
0718 69 1555 0659		J			69 E0001		BRING WORD SHOWING EQUIVALENT	1542
0765 60 1555 1088		0 800			60 A0201	TA		1772
0916 24 1555 0000	1555	0 0			24 A0201		0000 A0201	2434
1352 69 1555 0172		0 0			69 A0201	T0005	T0005 COMPARISON CONSTANT FOR END OF JOB	2604
1245 69 1555 0168		0 05			69 A0201	T0001	T0001 TO RESTORE BEFORE END OF JOB	2606
0365 20 1660 0000	1660	0 05A			20 G0001		0000 G0001 RESTORE COMPARISON CONSTANT AND MODIFY	982
0592 65 1660 0665		J			65 G0001		BRING EQUIVALENT OF REGION	1486
1266 24 1660 0029	1660	J ST			24 G0001		0029 G0001 STORE ADDRESS OF 0001 OF REGION	2492
0405 20 1661 0000	1661	0 0A			20 G0002		0000 G0002 COMPARISON CONSTANT	980
0647 24 1669 0072	1669				24 ORUMT		ORUMT PRESET DRUM TAG)	966
0370 69 1669 0772		R J0003			69 ORUMT		BLANK 1, IS ORUM FULL	1302
0611 69 1669 1072		N SUB11			69 ORUMT		IS THE DRUM FULL	1608
0617 24 1669 1122	1669				24 DRUMT		ORUMT CHANGE DRUM TAG TO 9)	1702
0420 69 1669 1224		R L0003			69 ORUMT		BLANK ADDRESS, IS ORUM FULL	2080
0616 69 1669 1274					69 ORUMT		BLANK 0 AND 1, CHECK DRUM TAG	2194
1298 24 1669 1280	1669				24 ORUMT		ORUMT CHANGE DRUM TAG, ORUM CANNOT BE FULL.	2448
0341 69 1679 0232		N 888			69 BLANB		SAVE IN CASE 1 IS BLANK	798
0438 24 1679 0282	1679				24 BLANB		BLANB SPEC. SUBR. BRING BACK BLANB + ORCEB	870
0268 20 1679 0271	1679	N M0001			20 BLANB	M0004	BLANB STORE EQUIVALENT OF BLANK ADDRESS	1180
1008 65 1679 0376					65 BLANB	A	BRING EQUIVALENT OF BLANK BACKWARD L	1238
0626 65 1679 0371					65 BLANB	J0004	BRING EQUIVALENT OF BLANK BACKWARD L	1306
1283 65 1680 0690		N 00NE			65 ZEROX		CLEAR ACCUMULATOR	2428
0381 69 1680 0533					69 ZEROX	RU	BRING ZERO TO INDICATE RESERVATION	1032
0220 65 1680 0177		N F0003			65 ZEROX	EXITX	ORUM ADDRESS, OR EQUIVALENT TO ORUM	1158
0392 24 1689 0442	1689	J SS			24 Z0001		Z0001 STORE SYMBOL)	1336
0290 10 1689 1002		J Y			10 Z0001	LOOP	A00 A SYMBOL INTO UPPER, FROM BOTTOM	1508
0340 10 1898 1002		J			10 ZMAXM	LOOP	A00 A SYMBOL, STARTING AT TOP	1516
0990 10 1898 0608		J			10 ZMAXM	U0001	A00 A SYMBOL TO UPPER	2500
0134 20 1899 0302	1899	J			20 ZMAX1		ZMAX1 CLEAR REGION AND SYMBOL TABLES)	954
0744 11 8001 0452					11 8001		DUPLICATE INTO AVAILABILITY TABLE)	2412
0634 11 8001 8002					11 8001	8002	DUPLICATE INTO AVAILABILITY TABLE)	2418
0070 65 8001 0327					65 8001		BRING BACK A SET)	640
0332 15 8001 0289					15 8001		BRING BACK A SET OF RESULTS)	918
0389 10 8001 0295					10 8001		MAKE SEVERAL VARIABLE ORDERS)	1060
0461 11 8001 0367					11 8001		MAKE SEVERAL VARIABLE ORDERS)	1066
0481 11 8001 0437					11 8001		MAKE SEVERAL VARIABLE ORDERS)	1072
0412 10 8001 0969					10 8001		RESTORE TO POSITIVE	1380
0467 10 8001 1223		N RC			10 8001		CORE ADDRESS. RESTORE THE 2000	1394
0542 10 8001 0713					10 8001	RES	RESTORE THE 9060	1402
0517 10 8001 1323		C NZ			10 8001		WRONG SYMBOL. RESTORE UPPER TO PLUS	1428
0947 11 8001 0604					11 8001		CLEAR FIRST LETTER	1446
0240 10 8001 0997					10 8001		ARE LAST FOUR CHARACTERS DIGITS)	1460
0679 65 8001 0535		N OUT			65 8001		THEY ARE DIGITS. GET VALUE)	1464
1063 67 8001 1319					67 8001	LR	EQUIVALENT IS IN 1 POSITION OF LOWER	1546
0314 67 8001 0721		N LH			67 8001		EQUIVALENT IN 0 POSITION OF LOWER	1548
0685 10 8001 1291					10 8001		AVAILABILITY WORD BACK INTO UPPER	1724
1035 15 8001 1341					15 8001		WHAT CELL OLD WE RESERVE)	1752
1270 15 8001 1227					15 8001		L-0 ADDENDS TO LEFT END LOWER)	1794
0986 16 8001 0993					16 8001		CLEAR OP FROM ACCUMULATOR	1936
0732 10 8001 0999		N ORM			10 8001	A	ORUM. MAKE POSITIVE AGAIN	2058
1285 60 8001 1292					60 8001		PUT IT INTO UPPER	2088
0694 60 8001 0709					60 8001	SUB10	PUNCH A CARD, AND REPEAT RBO	2408
0465 65 8002 0723					65 8002		REPLACE DIGIT OF AVAIL. WORD BY W)	1002
0683 66 8002 0491		J			66 8002		CHANGE SIGN OF CORRECTION	1258
0969 60 8002 0777					60 8002		NUMERIC ADDRESS ALONE IN UPPER	1382
0613 65 8002 0177		N FXT			65 8002	EXITX	FIXED ADDRESS. CLEAR UPPER AND EXIT.	1410

1323 65 8002 0731		65 8002		CLEAR UPPER	1430
0770 16 8002 0729		16 8002		ADDRESS INTO LEFT END DISTRIBUTOR	1562
0501 15 8002 0909		15 8002		4 TIMES DYNAMIC ADDRESS IN 0 POSITION	1626
0909 15 8002 0693		15 8002	SSW	4 TIMES DYNAMIC ADDRESS IN 0 POSITION	1628
1021 16 8002 1079		16 8002		WHAT CELL 010 WE RESERVE)	1748
1055 16 8002 1313		16 8002		(-0 ADDENDS TO LEFT END LOWER)	1790
0799 16 8002 0910		16 8002		0-ADDRESS TO LEFT END OF DISTR)	1836
1327 65 8002 0985		65 8002	BOTH	ODO. SEPARATE LAST ADDEND.	1896
1056 65 8002 0965		65 8002		DELETE 0 TAG	1942
0662 15 8002 1221		15 8002		DOUBLE IT	2046
1062 90 8002 1067		90 8002		0, WE ARE NOT ON FIRST BACKWARD CARO	2238
0022 65 8003 0229		65 8003	LOOP	BRING BACK A SET)	550
1080 65 8003 1038		65 8003		WHICH L IS LESS, MEASURED ON CIRCLE)	838
0212 16 8003 0669		16 8003		MODIFY TO END WITH 1999)	1052
0543 10 8003 0401		10 8003		MAKE SEVERAL VARIABLE ORDERS)	1076
0567 11 8003 0725	C LOO	11 8003		ARE LAST FOUR CHARACTERS DIGITS)	1452
1319 69 8003 0776	C LR	69 8003		CLEAR DISTRIBUTOR	1552
0689 60 8003 0057		60 8003	SB10A	CLEAR LOWER	1586
0736 65 8003 0693		65 8003	SSW	CLEAR ACCUMULATOR AND JUMP	1616
0667 11 8003 0975		11 8003		PUT INTO DISTRIBUTOR	1720
0717 60 8003 1225		60 8003		CLEAR LOWER	1730
1074 15 8003 1031		15 8003		WHAT CELL 010 WE RESERVE)	1758
0936 60 8003 0943		60 8003		SHOULD WE INTERCHANGE ADDENDS)	1862
1272 65 8003 0179		65 8003	EXITZ	NEW DYNAMIC LEVEL INTO LOWER	1916
1210 69 8003 0566		69 8003		CLEAR DISTRIBUTOR	1932
0648 69 8003 1261		69 8003		CLEAR DISTRIBUTOR	1968
0546 67 8003 0178	N NZ	67 8003	EXITY	IF NOT 8001, CORRECTION IS ZERO	2020
0933 11 8003 1242		11 8003		CORE. CLEAR UPPER	2054
0716 61 8003 0780		61 8003		CHANGE ITS SIGN	2396
0955 21 9972 0755	9972 J STR	21 9972	SKP	9972 NEW AVAILABILITY'WORO INTO ORUM)	1738
0360 69 9999 8003	J BRNG	69 9999	8003	BRING BACK A SET)	644
0074 69 9999 8003	J BRNG	69 9999	8003	BRING BACK A SET OF RESULTS)	924

INSTRUCTIONS LISTED IN ORDER OF LOCATION ADDRESS

0001	60	0003	0709	N	SETCC	60	REAO	SUB10	SEQUEL TO MANY PSEUDO-OPS.	1580
0002	69	1112	0039	R	0002	69		SUB14	OP, C.I., TAGS, OPTIM, ALOPT	2248
0003	70	0161	1041		REAO	70	R0011		REAO ONE CARO	364
0004	90	0088	0000	K	P	90	0088	0000	TAG FOR SEARCH STARTED BY MOF	520
0005	69	0208	0061			69		SUBR3	IS L FIXED ADDRESS)	440
0006	24	0178	0231	0178	N	SUBR1	24	EXITY	STORE EXIT	950
0007	44	0011	0012				44	MOFLI	IF SO, START FORWARD SEARCH	388
0008	69	1044	0047	N	0008	69	81STX		DELETE QUITT TAG)	372
0009	65	0553	0907	J		65	VAR		IS THIS LAST SET)	662
0010	69	1234	1244	J	0010	69	85TH		C.I. FOR AVAILABILITY TABLE)	2552
0011	69	0014	0107	N	MOFLI	69	I	A	STARTED SEARCH WITH INOEXEO O	400
0012	69	0015	0018	C	ABCOE	69		SUB21	PROCESS I	390
0013	22	0167	0070	0167			22	XXXX1	BRING BACK A SET)	638
0014	90	0089	0000	K	I	90	0089	0000	TAG FOR SEARCH STARTED BY INOEXEO O	522
0015	60	0003	0057	J		60	REAO	SB10A	PREPARE TO PUNCH CARO	392
0016	65	0114	0984	R	0016	65	91STX		TAG IN CASE L IS UNDEFINITE SYMBOL)	2522
0017	24	1050	0253	1050		24	1050		CHANGE FIRST-CARO INOICATOR TO 2NO)	490
0018	24	0177	0680	0177	N	SUB21	24	EXITX	STORE EXIT	2178
0019	69	1050	0503		N	BACKW	69	1050	WAS THERE JUST ONE SEARCH CARO	604
0020	69	0182	0085				69	P0006	SAVE L FROM CARO THAT STARTED SRCH)	622
0021	69	1044	1298	R	0021	69	81STX		BLA ENTRY. CHANGE ORUM TAG	2446
0022	65	8003	0229			65	8003	LOOP	BRING BACK A SET)	550
0023	45	0026	0077			45	QUITT		O, IT IS NOT, AND WE QUIT	436
0024	69	0186	8002	J		69	P0010	8002	STORE PROCESSED CARO)	880
0025	60	0153	0764	R	0025	60	R0003	BOTH	SYN ENTRY. ALPHABETIC I	2314
0026	69	1050	0303	N	QUITT	69	1050		WAS THERE ONLY ONE SEARCHED CARO	532
0027	69	0001	0006	R	0027	69	SETCC	SUBR1	TO INITIALIZING SUBROUTINE	2638
0028	11	0031	0035	X		11	O		STORE SET OF OATA)	414
0029	69	1232	0740	R	0029	69	ZO	SUB	BLR ENTRY. TO SUBROUTINE	2442
0030	69	0633	0063			69		SUBR9	TO SUBROUTINE 9 TO ANALYZE THE ADDRESS	1150
0031	69	0155	8002	Q	O	69	R0005	8002	COMPARISON CONSTANT FOR STORING	514
0032	65	0103	0407	N	BCOEF	65	FINAL		IS STORAGE AREA FULL)	478
0033	69	0036	0039			69		SUB14	OP, C.I., OPTIM, ALOPT, ETC.	376
0034	99	0338	0239	N	XX8	99		9X8	MULTIPLE BRANCH ACCORDING TO TAGS)	744
0035	15	0038	0043			15	16THX		STORE SET OF OATA)	416
0036	69	0089	0042	J		69		SUB19	PROCESS L	378
0037	24	0151	0104	0151		24	R0001		R0001 IF SO, PUT IN DUMMY MOVEABLE WORDS)	504
0038	00	0001	0000	K	16THX	00	0001	0000		2668
0039	24	0177	0430	0177	N	SB14	24	EXITX	EXITX STORE EXIT	1924
0041	10	0044	8003	C	LOOP	10		8003	STORE SET OF OATA)	408
0042	24	0177	0480	0177	N	SUB19	24	EXITX	EXITX STORE EXIT	2068
0043	44	0147	0148			44		OUT	STORE SET OF OATA)	418
0044	69	0151	8002	J		69	R0001	8002	STORE SET OF OATA)	410
0045	24	0177	0630	0177	N	SUB20	24	EXITX	EXITX STORE EXIT	2116
0046	60	0152	0257			60	R0002		IF SO, IS IT FIXED)	452
0047	24	1030	0033	1030		24	1030		DELETE QUITT TAG)	374
0048	94	0051	0053			94	MOFLP		O, YES	384
0049	24	0052	0205	0052		24	DONE		OONE PRESET EXIT FROM BACKWARDS ROUTINE)	632
0050	69	0156	8002	O	OS	69	R0006	8002	TO RESTORE COMPARISON CONST + ADVANCE	516
0051	69	0004	0107	N	MOFLP	69	P	A	STARTED SEARCH WITH MOF OPERATION	398
0053	60	0902	0007			60	0902		IS O INOEXEO	386
0054	60	0153	0764	R	0054	60	R0003	BOTH	EOU ENTRY. ALPHABETIC I	2312
0055	60	0902	0307			60	0902		IS O INOEXEO	464
0056	69	1050	0203			69	1050		IS THIS FIRST SET OF SEARCH)	424
0057	15	0181	0635	C	SB10A	15	P0005		ADVANCE CARO NUMBER)	1588
0058	65	0160	0065	N	NO	65	R0010		IS CARO OF TYPE 08)	432
0059	20	0197	0500	0197		20	ALOFT		STORE MODIFIED ALOPT	810
0060	44	0026	0064	J		44	QUITT		IF N G, QUIT SEARCH	456
0061	24	0177	0030	0177	N	SUBR3	24	EXITX	EXITX STORE EXIT	1148
0063	24	0179	0482	0179	N	SUBR9	24	EXITZ	EXITZ STORE EXIT	1434
0064	45	0144	0019			45	K	BACKW	IF FIXED, START BACKWARD PROCESSING	458
0065	16	0068	0023			16	810TH		IS CARO OF TYPE 08)	434
0066	65	0197	0351	J		65	ALOFT		MODIFY ALOPT AGAIN)	854
0067	24	1050	0358	1050		24	1050	A	1050 START ON L. FIRST-CARO TAG)	616
0068	00	0000	0008	K	810TH	00	0000	0008		2650
0070	65	8001	0327			65	8001		BRING BACK A SET)	640
0071	44	0026	0076			44	QUITT		IF SO, QUIT SEARCH)	444
0072	65	0175	0429			65	AVAL		MAKE DRUM AVAILABLE)	968
0073	60	0010	0709	R	0073	60	0010	SUB10	PUNCH USUAL DUMMY OUTPUT CARO	2550
0074	69	9999	8003	J	BRNG	69	9999	8003	BRING BACK A SET OF RESULTS)	924
0075	00	1000	0000	K	13RO	00	1000	0000		754
0076	45	0088	0026			45	C	QUITT	IF SO, QUIT SEARCH)	446
0077	60	0151	0005			60	R0001		ALPHABETIC L	438
0078	45	0032	0019			45	BCOEF	BACKW	IF I IS FIXED, START BACKWARD	476
0079	00	0000	0007	K	710	00	0000	0007		526
0080	60	0151	0455			60	R0001		ALPHABETICAL L INTO UPPER	1174
0081	91	0034	0436	C	F	91	XX8		MULTIPLE BRANCH ACCORDING TO TAGS)	738
0082	24	0151	0204	0151	J	SEND	24	R0001	BRING BACK A SET)	556
0083	65	0086	0041			65	SENO	X	STORE SET OF OATA)	406
0084	65	0990	8002	R	0084	65		8002	ADJOIN ORDER TO LOWER	2498
0085	24	0238	0191	0238		24	SAVEL		SAVE L FROM CARO THAT STARTED SRCH)	624
0086	24	1125	0028	1125	J	SEND	24	S0001	STORE SET OF OATA)	412
0087	24	0040	0193	0040		24	SAVEO	X	SAVE O FROM CARO THAT STARTED SRCH)	628
0088	69	0091	0039	C	C	69		SUB14	GET OP, C.I., TAGS, OPTIM, ALOPT.	448
0089	69	0092	0045	J		69		NO	PROCESS O	380
0090	24	0052	0468	0052		24	OONE		WITH L. ALTER EXIT)	684
0091	95	0144	0046	J		95	K	OONE	IS O ACTUALLY AN ADDRESS	450
0092	69	0095	0048	J		69	OPTIM		IS IT AN MOF OPERATION)	382
0093	65	0152	1345	R	0093	65	R0002		ALPHABETIC O	2620
0094	60	0158	1114	R	0094	60	R0008		BEGINNING OF REGION TO BE RESERVED	2380
0096	65	0117	0321	J	DN1	65	VAR2X		ALL CASES. STORE PROCESSED CARO)	876

0097	65	0152	1116	R	0097	65	R0002		REGION DESIGNATOR TO RIGHT END)	2474
0098	65	0159	0964	R	0098	65	R0009		REQ ENTRY. NUMERICAL I	2362
0099	69	0502	0740	R	0099	69		SUB	RBR ENTRY. TO SUBROUTINE	2454
0100	69	0003	0006	N	0100	69	READC	SUBR1	INITIALIZE AT START OF ASSEMBLY	362
0101	65	0397	0201	J		65	LNM	SUB22	SAVE ORCEO	776
0102	65	0052	0201	J LN		65	00NE	SUB22	SAVE ORCEO	786
0104	24	0152	0105		0152	24	R0002		R0002 IF SO, PUT IN DUMMY MOVEABLE WORDS)	506
0105	24	0153	0236		0153	24	R0003	YYY	R0003 IF SO, PUT IN DUMMY MOVEABLE WORDS)	508
0106	16	0069	0623			16	OC		IS THIS LAST SET	568
0107	24	1030	0083		1030	24	1030		1030 STORE TAG TELLING HOW SEARCH STARTED	404
0109	65	0062	0217	C A		65	OPREG		SAVE DYNAMIC LEVEL OF L FROM 0)	818
0110	35	0005	0173	J		35	0005		IS I A FIXED ADDRESS)	472
0111	69	0114	0017			69	91STX		CHANGE FIRST-CARD INOICATOR TO 2ND)	488
0112	11	0365	8003			11	OSA	8003	CLEAR REGION AND SYMBOL TABLES)	960
0113	69	0235	0438	N SSB		69	T1		SPEC. SUBR. BRING BACK BLANK + ORCEB	868
0114	90	0000	0000	K 91STX		90	0000	0000		2678
0115	45	0318	0026			45		QUIT	IFSO, QUIT SEARCH	482
0116	20	1551	0900	1551	Q 796	20	A0197	0900	A0197 COMPARISON CONSTANT FOR LAST LINE	1132
0118	65	1328	1233		R 00001	65	83RD	BP	INVALID ADDRESS	2320
0119	65	1328	1233		R 00002	65	83RD	BP	BLANK I. CARD IS UNUSABLE	2322
0120	20	0095	1098	0095	N 00003	20	E	X	E DRUM. PUT I-EQUIVALENT INTO E	2324
0121	65	1328	1233		R 00004	65	83RD	BP	NEW SYMBOL I. CARD IS UNUSABLE	2326
0122	20	0095	1098	0095	R 00005	20	E	X	E 800X ADDRESS	2328
0123	20	0095	1098	0095	R 00006	20	E	X	E CORE ADDRESS	2330
0124	20	0095	1098	0095	R 00007	20	E	X	E OTHER ADDRESS	2332
0125	65	1328	1233		R 00008	65	83RD	BP	N G O. CARD IS UNUSABLE	2340
0126	65	1328	1233		R 00009	65	83RD	BP	BLANK D. CARD IS UNUSABLE	2342
0127	20	0536	1289	0536	N 00010	20	EQUIV	SD	EQUIV OLD SYMBOL. STORE I AS NEW EQUIV	2344
0128	20	0536	1289	0536	R 00011	20	EQUIV	SD	EQUIV NEW SYMBOL. STORE I AS ITS EQUIVALENT	2346
0129	20	0536	1289	0536	R 00012	20	EQUIV	SD	EQUIV 800X. STORE ITS EQUIVALENT.	2348
0130	20	0536	1289	0536	R 00013	20	EQUIV	SD	EQUIV CORE. STORE I AS ITS EQUIVALENT.	2350
0131	20	0536	1289	0536	R 00014	20	EQUIV	SD	EQUIV OTHER. STORE I AS THE EQUIVALENT.	2352
0132	01	0000	0000		K 12NDX	01	0000	0000		2672
0133	96	0136	0088			96	B	C	WITH I OR L, RESPECTIVELY	430
0134	20	1899	0302	1899	J	20	ZMAX1		ZMAX1 CLEAR REGION AND SYMBOL TABLES)	954
0135	69	0040	0343			69	SAVED		BRING BACK L AND D)	678
0136	60	0153	0357		C B	60	R0003		IS I A FIXED ADDRESS)	468
0137	65	0117	0571	J ENDED		65	VAR2X		MAKE NEW BRINGING ORDER)	904
0138	69	0141	0042			69		SUB19	PROCESS L	540
0139	69	1044	0297	J		69	81STX		CHANGE TAG TO SECOND-CARD TAG)	588
0140	98	0291	0145			98	MISC	899	MULTIPLE BRANCH ACCORDING TO TAGS)	742
0141	69	0012	0045	J		69	ABCOE	SUB20	PROCESS D, AND TO 08 ROUTINE	542
0142	24	0177	0330	0177	N SUBR6	24	EXITX		STORE EXIT	1290
0143	44	0247	0198			44		OUT	BRING BACK A SET)	562
0144	69	0197	0150		C K	69	ALOPT		IS IT AN MDF OPERATION	460
0145	69	0298	0400		N 899	69		SUBR5	PROCESS D BACKWARDS	772
0146	24	0155	0408	0155	J	24	R0005	X	R0005 BRING BACK A SET)	648
0147	10	0050	8003			10	QS	8003	STORE SET OF DATA)	420
0148	20	0103	0056	0103	N OUT	20	FINAL		FINAL STORE VARIABLE ORDER	422
0149	69	0102	0018	J		69	LN	SUB21	PROCESS I FORWARDS	784
0150	94	0026	0055			94	QUIT		IF SO, QUIT SEARCH	462
0161	15	0038	0143			15	16THX		BRING BACK A SET)	560
0162	16	0515	0619		J	16	110TH		REDUCE NUMBER YET TO BE DONE)	1014
0163	69	0066	0400			69		SUBR5	PROCESS BACKWARDS D ONCE MORE	852
0164	24	0117	0020	0117		24	VAR2X		PRESET TO STORE AFTER PROCESSING)	620
0165	16	0068	0323			16	810TH		IS IT TYPE 08)	496
0166	10	1003	0558			10	N		UPPER, LAST ADDRESS - 2000	1048
0168	24	0178	0945	0178	R T0001	24	P0002		STORE AVAILABILITY WORD	2574
0169	24	0180	0995	0180	R T0002	24	P0004		STORE AVAILABILITY WORD	2578
0170	28	0182	1045	0182	R T0003	24	P0006		STORE AVAILABILITY WORD	2582
0171	28	0184	1095	0184	R T0004	24	P0008		STORE AVAILABILITY WORD	2586
0172	71	0177	1096		R T0005	71	P0001		PUNCH A CARD OF THE TABLE	2590
0173	44	0026	0078			44	QUIT		IF I IS N G, QUIT SEARCH	474
0174	69	1199	8003	O QL		69	SMAXM	8003		944
0175	11	1111	1111	K AVAL		11	1111	1111	AVAILABILITY WORD	988
0176	15	0079	0233			15	710		IS TYPE 01	500
0187	20	0197	0081	0197	N X2	20	ALOPT	F	ALOPT STORE MODIFIED ALOPT	736
0188	69	0114	0067		N B	69	91STX		START ON L. FIRST-CARD TAG)	614
0189	69	0238	0241	J		69	SAVEL		BRING BACK L AND D)	674
0190	00	0908	0344	J		00	0908		TAGS AND EXIT FOR SUBROUTINE 13	1270
0191	69	0184	0087			69	P0008		SAVE D FROM CARD THAT STARTED SRCH)	626
0192	69	0195	0305	J		69		SUBR4	PROCESS L BACKWARDS	792
0193	69	0096	0049			69	DN1		PRESET EXIT FROM BACKWARDS ROUTINE)	630
0194	69	0397	0400	J		69	LNM	SUBR5	PROCESS D BACKWARDS	760
0195	69	0102	0045	J		69	LN	SUB20	PROCESS D FORWARDS	794
0196	69	0149	0305	J		69		SUBR4	PROCESS L BACKWARDS	782
0198	20	0353	0106	0353	N OUT	20	VAR	VAR	STORE VARIABLE ORDER	566
0199	16	0202	1107			16	251XX	BOT	WHICH L IS LESS, MEASURED ON CIRCLE)	844
0200	24	0156	0204	0156	O OS	24	R0006	X	R0006 TO RESTORE AND ADVANCE	598
0201	69	1050	1062		N SUB22	69	1050		FIRST-CARD TAG	2236
0202	00	0000	0025	K 251XX		00	0000	0025		2652
0203	90	0207	0058			90		NO	IS THIS FIRST SET OF SEARCH)	426
0204	11	0607	0161		X	11	Q		BRING BACK A SET)	558
0205	65	0103	0757			65	FINAL		BRING BACK A SET)	634
0206	65	0353	0229	J		65	VAR	LOOP	BRING VARIABLE ORDER, AND REPEAT	594
0207	69	1030	0133			69	1030		IF SO, WHERE DID SEARCH START	428
0208	35	0005	0071	J		35	0005		IS L FIXED ADDRESS)	442
0209	20	0197	0397	0197		20	ALOPT	LNM	STORE NEW ALOPT, AND JUMP BACK	860
0210	28	1200	0028	1200	Q MAX	24	SMAX1	X	SMAX1 CONSTANT FOR WHETHER STORAGE IS FULL	518
0211	24	1199	0252	1199	J SEN	24	SMAXM		STORE PROCESSED CARD)	882
0212	16	8003	0669			16	8003		MODIFY TO END WITH 1999)	1052

0213	20	0181	0234	0181	N OUT	20	P0005	P0005	CLEAR CARD NUMBER	962
0214	69	0317	0142		J	69		SUBR6	PROCESS I BACKWARDS AGAIN	864
0215	22	0069	0022	0069		22	OC	OC	NOT 1ST CARO. MAKE COMP. CONST.)	548
0216	15	0769	0973			15	C5		MODIFY TO TAKE NEXT COLUMN OF TABLE)	1100
0217	14	0320	1036			14	501XX		SAVE DYNAMIC LEVEL OF L FROM D)	820
0218	60	0068	0177		R F0001	60	810TH	EXITX	ADDRESS IS N G	1154
0219	65	0068	0177		R F0002	65	810TH	EXITX	BLANK ADDRESS	1156
0220	65	1680	0177		N F0003	65	ZEROX	EXITX	ORUM ADDRESS, OR EQUIVALENT TO ORUM	1158
0221	65	0068	0177		R F0004	65	810TH	EXITX	SYMBOLIC ADDRESS, WITH UNDEFINED SYMB	1160
0222	35	0005	0177		R F0005	35	0005	EXITX	800X ADDRESS	1162
0223	65	0068	0177		R F0006	65	810TH	EXITX	CORE ADDRESS	1164
0224	60	0068	0177		R F0007	60	810TH	EXITX	OTHER ADDRESS. USUALLY PART OF CONST.	1166
0225	65	0359	0413		N TP	65	SPR		BOTTOM LINE OF TABLE	1094
0226	69	0279	0039			69		SUB14	GET OP, C.I., TAGS, OPTIM, ALOPT.	572
0227	65	0103	0041		C LP	65	FINAL	LOOP	ALL CASES, BACK TO START OF LOOP	512
0229	10	0082	8002		C LOOP	10	SENO	8002	BRING BACK A SET)	552
0230	60	0152	0958			60	R0002		ALPHABETIC D INTO UPPER	1224
0231	61	0134	8003			61		8003	CLEAR REGION AND SYMBOL TABLES)	952
0232	24	0235	0388	0235		24	T1	T1	SAVE IN CASE I IS BLANK	800
0233	45	0236	0037			45	YYY		O, NO, I, YES.	502
0234	69	1044	0647			69	81STX		PRESET ORUM TAG)	964
0236	24	0155	0227	0155	C YYY	24	R0005	LP	R0005 EITHER CASE, SOMETHING INTO R0005	510
0237	20	0197	0550	0197		20	ALOFT		MODIFY TO USE ROUTINE FOR UNFIXED D)	828
0239	69	0192	0142		N 9X8	69		SUBR6	PROCESS I BACKWARDS	790
0240	10	8001	0997			10	8001		ARE LAST FOUR CHARACTERS DIGITS)	1460
0241	24	0182	0135	0182		24	P0006		BRING BACK L AND O)	676
0242	24	0182	0285	0182	J SEND	24	P0006	X	BRING BACK A SET OF RESULTS)	926
0243	10	0146	8002		N LOOP	10		8002	BRING BACK A SET)	646
0244	24	0447	0450	0447		24	T2	T2	SAVE IN CASE I IS CORE)	804
0245	46	0348	0199			46	AE8		WHICH L IS LESS, MEASURED ON CIRCLE)	842
0246	35	0000	0750		P C1	35	0000	XX	TO PRESET LEFT SHIFT ORDER	1124
0247	10	0200	8002			10	05	8002	BRING BACK A SET)	564
0248	20	0553	0256	0553	N OUT	20	VAR	VAR	STORE VARIABLE ORDER	658
0249	22	0901	0354	0901		22	0901	0901	MAKE SEVERAL VARIABLE ORDERS)	1086
0250	24	0150	0408	0150	O OS	24	R0000	X	R0000 RESTORE COMPARISON CONSTANT AND MODIFY	750
0251	16	0254	0059			16	11STX		CHANGE ITS FIRST DIGIT TO 8	808
0252	11	0355	0259			11	02		STORE PROCESSED CARD)	884
0253	70	0151	1007			70	R0001		READ ANOTHER CARD	492
0254	10	0000	0000		K 11STX	10	0000	0000		2674
0255	60	0153	0957			60	R0003		BRING ALPHABETIC I	690
0256	69	0009	0039			69		SUB14	GET OP, C.I., TAGS, OPTIM, ALOPT.	660
0257	69	0060	0061			69		SUBR3	IF SO, IS IT FIXED)	454
0258	65	0103	0557		N SKP	65	FINAL		NOT 1ST CARO. MAKE COMP. CONST.)	544
0259	16	0038	0443			16	16THX		STORE PROCESSED CARD)	886
0260	69	1125	8003		J BRNG	69	S0001	8003	BRING BACK A SET)	554
0261	24	0151	0408	0151	O O	24	R0001	X	R0001 COMPARISON CONSTANT FOR END OF SET	748
0262	10	0565	0331			10	C3	SU	MAKE SEVERAL VARIABLE ORDERS)	1092
0263	11	0116	0671			11	796		ARE WE TO LAST LINE OF TABLE)	1020
0264	45	0000	0001		P C4	45	0000	0001	DIFFERENCE OF BRINGING AND STORING ORD	1130
0265	16	0038	0293			16	16THX		BRING BACK A SET)	652
0266	00	0196	0000		P C6	00	0196	0000	TO MODIFY BRINGING ORDER FOR NEXT COL	1138
0267	14	0320	1080			14	501XX		WHICH L IS LESS, MEASURED ON CIRCLE)	836
0268	20	1679	0271	1679	N M0001	20	BLANB	M0004	STORE EQUIVALENT OF BLANK ADDRESS	1180
0269	60	0622	0627		R M0002	60	87THX	S8108	N. G. OMIT PUNCHING	1182
0270	69	1044	0697		R M0003	69	81STX		BLANK. CHANGE FIRST-CARO TAG)	1184
0271	20	0182	0335	0182	N M0004	20	P0006	SB10C	P0006 DRUM ADDRESS. STORE FOR PUNCHING	1192
0272	65	0325	0611		R M0005	65		SUB11	NEW SYMBOL. FIND BEST EQUIVALENT	1194
0273	01	0555	0273		R M0006	01	0555	M0006	800X ADDRESS. STOP THE MACHINE	1198
0274	20	0182	0385	0182	R M0007	20	P0006	P0006	CORE ADDRESS. STORE FOR PUNCHING	1200
0275	01	0555	0275		R M0008	01	0555	M0008	OTHER ADDRESS. STOP MACHINE	1208
0276	69	0197	0950		N M0009	69	ALOFT		EQUIV OF NEW SYMBOL.	1210
0277	00	0000	0269		R M0010	00	0000	M0002	NEW SYMBOL, BUT SYMBOL TABLE FULL	1216
0278	35	0001	0485			35	0001		ARE LAST FOUR CHARACTERS DIGITS)	1456
0279	69	1050	0403		J	69	1050		IS THIS FIRST OUITT SET	574
0280	69	0683	0586			69		SUB17	CORRECTION TO DYNAMIC LEVEL	1256
0281	30	0001	0337			30	0001		REPLACE DIGIT OF AVAIL. WORD BY W)	1006
0282	69	0447	0600			69	T2		SPEC. SUBR. BRING BACK BLANB + ORCEB	872
0283	96	0012	0138			96	ABCOE		IF WITH I, PROCESS LIKE 08	538
0285	11	0488	0493		X	11	0		BRING BACK A SET OF RESULTS)	928
0286	69	0139	0018		J JUMP	69		SUB821	PROCESS I	586
0287	15	0254	0209		C XY	15	11STX		MODIFY ALOPT AGAIN)	858
0288	69	0137	0090		N L	69	ENDED		WITH L. ALTER EXIT)	682
0289	10	0242	0547			10	SEND	A	BRING BACK A SET OF RESULTS)	920
0290	10	1689	1002		J Y	10	Z0001	LOOP	ADD A SYMBOL INTO UPPER, FROM BOTTOM	1508
0291	69	0194	0142		N MISC	69		SUBR6	PROCESS I BACKWARDS	758
0293	44	0347	0248			44		OUT	BRING BACK A SET)	654
0295	30	0003	1053			30	0003		MAKE SEVERAL VARIABLE ORDERS)	1062
0296	15	0549	0954			15	601		INITIAL OF VARIABLE BRINGING ORDER	1638
0297	24	1050	0453	1050		24	1050		CHANGE TAG TO SECOND-CARO TAG)	590
0298	69	0101	0018		J	69		SUB821	PROCESS I FORWARDS	774
0299	22	0901	0404	0901		22	0901	0901	MODIFY TO TAKE NEXT COLUMN OF TABLE)	1104
0300	95	0603	0255			95	SHX		O, NO, I, YES.	688
0301	15	0132	0237			15	I2NDX		MODIFY TO USE ROUTINE FOR UNFIXED D)	826
0302	10	0405	0309			10	0A		CLEAR REGION AND SYMBOL TABLES)	956
0303	90	0507	0258			90		SKP	O, ONLY ONE.	534
0305	24	0177	0080	0177	N SUBR4	24	EXITX	EXITX	STORE EXIT	1172
0306	69	0397	0142		J	69	LNM	SUBR6	PROCESS I BACKWARDS	768
0307	45	0026	0136			45	QUITT	B	IF SO, QUIT SEARCH	466
0308	69	1030	0333			69	1030		IF SO, WHERE DID WE START	578
0309	46	0112	0213			46		OUT	CLEAR REGION AND SYMBOL TABLES)	958
0310	69	0286	0045		J	69	JUMP	SUB20	PROCESS D	584

0311	65	0214	0113	N	2N0	65		SSB	USE I-COMP. BRING BLANK AND ORCEB	862
0312	60	0108	0513			60	SA		MODIFY TO TAKE FIRST LINE, SAME COLUMN	1108
0313	46	0166	0178			46		EXITX	I, OVER 1999. WE RESERVE NOTHING	1046
0314	67	8001	0721	N	LH	67	8001		EQUIVALENT IN D POSITION OF LOWER	1548
0315	45	0468	0319			45	NO		D, NO, I, YES.	666
0316	00	0195	0000	P	C8	00	0195	0000	MODIFY BRINGING ORDER TO START NEW COL	1142
0317	65	0197	0287	J		65	ALOFT	XY	BRING ALOFT, AND JUMP BACK	866
0319	65	0903	0457			65	0903		IS I INDEXED	484
0319	69	1030	0433			69	1030		IF SO, WHERE OLD SEARCH START	668
0320	00	0000	0050	K	50IXX	00	0000	0050		2656
0321	10	0024	8003			10		8003	STORE PROCESSED CARD)	878
0322	20	1355	0000	1355	O	05B	20	A0001	RESTORE COMPARISON CONSTANT AND MODIFY	986
0323	45	0176	0227			45		LP	O, IT IS NOT. I, IT IS.	498
0324	10	0427	0331			10	800	SU	RESTORE, AND TAKE WORD FROM NEXT LINE	1024
0325	00	0989	0276	J		00	0989	M0009	TAGS AND EXIT FOR USE IN SUBROUTINE 11	1196
0327	16	0038	0243			16	16THX	LOOP	BRING BACK A SET)	642
0328	35	0004	0639			35	0004		MAKE BRINGING ORDER)	1482
0329	20	0197	0350	0197	C	X1	20	ALOFT	STORE MODIFIED ALOFT	712
0330	60	0153	1108			60	R0003		ALPHABETIC I TO UPPER	1292
0331	21	0108	0361	0108	C	SU	21	SA	STORE VARIABLE STORING ORDER	1026
0332	15	8001	0289			15	8001		BRING BACK A SET OF RESULTS)	918
0333	96	0286	0657			96	JUMP	NO	D, STARTED WITH I. I, WITH L.	580
0334	00	0000	9060	K	9060	00	0000	9060		1414
0335	20	0062	0177	0062	N	SB10C	20	OPREG	OPREG FROM 4-5, 19-20. STORE DYNAMIC LEVEL	1602
0336	95	0032	0012			95	BCOEF		WITH I. EXIT ACCORDING TO WHY SEARCH	612
0337	10	0304	0359			10	UH		REPLACE DIGIT OF AVAIL. WORD BY W)	1008
0338	98	0341	0393			98	888		MULTIPLE BRANCH ACCORDING TO TAGS)	746
0339	24	0292	0003	0292	J		24	ORCEB	FINISHED. SAVE SAVOR AS ORCEQ)	938
0340	10	1898	1002			10	ZMAXM	LOOP	ADD A SYMBOL, STARTING AT TOP	1516
0341	69	1679	0232		N	888	69	BLANK	SAVE IN CASE I IS BLANK	798
0342	20	0391	0335	0391			20	ORCEB	STORE DYNAMIC LEVEL OF CORE ADDRESS	1206
0343	24	0184	0137	0184			24	P0008	BRING BACK L AND O)	680
0344	69	1050	0504			69	1050		FIRST-CARD TAG	1272
0345	20	0435	0588	0435			20	LSYMB	STORE ADDR OF SPACE RELATIVE TO START	1520
0347	10	0250	8002			10	QS	8002	BRING BACK A SET)	656
0348	15	0202	1107		N	AEB	15	25IXX	WHICH L IS LESS, MEASURED ON CIRCLE)	846
0349	24	0901	0454	0901			24	0901	LAST LINE AND END OF WORD. MODIFY)	1114
0350	60	0152	1057			60	R0002		ALPHABETIC O, TO SEE WHETHER FIXED	714
0351	16	0132	0287			16	12NOX	XY	MODIFY ALOFT AGAIN)	856
0352	60	0515	1069			60	110TH	00N	MODIFY EXIT, AND PREPARE TO OUIT	1528
0354	69	0708	0511			69	C2		MAKE SEVERAL VARIABLE ORDERS)	1088
0355	69	0182	8002	Q	O2		69	P0006	COMPARISON CONSTANT FOR END OF SET	896
0356	65	0109	0113	J		65		SSB	BRING BACK BLANK AND ORCEB	816
0357	69	0110	0061			69		SUBR3	IS I A FIXED ADDRESS)	470
0358	69	0211	0164	C	A		69	SEN	PRESET TO STORE AFTER PROCESSING)	618
0360	69	9999	8003	J	BRNG		69	9999	BRING BACK A SET)	644
0361	10	0264	8003			10	C4	8003	MAKE VARIABLE BRINGING ORDER	1028
0362	60	0108	0563			60	SA		LAST LINE AND END OF WORD. MODIFY)	1120
0363	15	0228	0583			15	P		MAKE SEVERAL VARIABLE ORDERS)	1080
0364	99	0688	0490			99	FD	FIBD	I, BACKWARD L, BRANCH, UNFIXED D	1814
0365	20	1660	0000	1660	Q	QSA	20	G0001	RESTORE COMPARISON CONSTANT AND MODIFY	982
0366	10	1019	1123			10	0DIFX	0000	DRUM, ADD RELOCATION AMOUNT	1388
0367	35	0001	0773			35	0001		MAKE SEVERAL VARIABLE ORDERS)	1068
0368	69	0911	0661		N	J0001	69	PRE	STORE EQUIVALENT OF NEW SYMBOL	1298
0369	60	0722	0627	R	J0002		60	89THX	N.G. OMIT PUNCHING	1300
0370	69	1669	0772	R	J0003		69	ORUMT	BLANK I, IS DRUM FULL	1302
0371	24	0062	0615	0062	N	J0004	24	OPREG	DRUM OR DRUM EQUIVALENT	1308
0372	65	0675	0611	R	J0005		65		NEW SYMBOL. FIND BEST VALUE	1314
0373	01	0000	0373	R	J0006		01	0000	800X RANGE. ERROR.	1318
0374	69	0391	0371	R	J0007		69	ORCEB	CORE. BRING DYNAMIC LEVEL	1320
0375	20	0185	0177	0185	R	J0008	20	P0009	OTHER ADDRESS, USUALLY A CONSTANT	1322
0376	20	0062	0925	0062	N	A	20	OPREG	OPREG STORE ADDRESS FOR OPTIMIZING NEXT ADDR	1278
0377	22	0167	0620	0167			22	XXXX1	MAKE NEW BRINGING ORDER)	908
0378	60	0000	0605			60	0000		BRING APPROPRIATE TABULAR VALUE	1568
0379	45	0332	0483			45		OVR	BRING BACK A SET OF RESULTS)	916
0380	11	0254	0559			11	11STX		IS FIRST POSITION BLANK	1376
0381	69	1680	0533			69	ZEROX	RU	BRING ZERO TO INDICATE RESERVATION	1032
0382	20	1554	1307	1554	J		20	A0200	MAKE DRUM AVAILABLE)	972
0383	96	0336	0188			96		B	D, WITH I. I, WITH L.	610
0385	69	0538	0441			69		SUB13	FIND DYNAMIC LEVEL OF CORE ADDRESS	1202
0386	69	0189	0142			69		SUBR6	PROCESS I BACKWARD	672
0387	20	1003	0506	1003			20	N	STORE N-1	1042
0388	69	0391	0244			69	ORCEB		SAVE IN CASE I IS CORE)	802
0389	10	8001	0295			10	8001		MAKE SEVERAL VARIABLE ORDERS)	1060
0390	22	0167	0441	0167	J	SS	22	XXXX1	PREPARE EXIT AND GO TO SUBROUTINE 13)	1620
0392	24	1689	0442	1689	N	SS	24	Z0001	STORE SYMBOL)	1336
0393	69	0196	0400		N	898	69		PROCESS D BACKWARDS	780
0394	23	1555	0179	1555	P	P3	23	E0001	INITIAL OF ORDER TO STORE I-POSITION	1368
0395	00	0210	0000	K	ZTABL		00	0210	LENGTH OF SYMBOL TABLE	2680
0396	15	0599	1104			15	602		INITIAL BRINGING ORDER)	1662
0397	69	0052	0305	J	LNK		69	00NE	PROCESS L BACKWARDS	762
0398	20	0117	0570	0117	N	OUT2	20	VAR2X	STORE PROCESSED CARD)	892
0399	44	0654	0704			44		ABS	I, ZERO. ADDRESS WAS ABSOLUTE	1472
0400	24	0177	0230	0177	N	SUBR5	24	EXITX	STORE EXIT	1222
0401	10	0658	0363			10	X		MAKE SEVERAL VARIABLE ORDERS)	1078
0402	60	0184	0789		N	SHOP	60	P0008	SHIFT OPERATION. FORWARD I, BACKW L.	1816
0403	90	0657	0308			90	NO		O, IT IS NOT	576
0404	20	0359	0312	0359			20	SPR	MODIFY TO TAKE NEXT COLUMN OF TABLE)	1106
0405	20	1661	0000	1661	Q	OA	20	G0002	COMPARISON CONSTANT	980
0406	65	0062	0267	J		65	OPREG	0000	WHICH L IS LESS, MEASURED ON CIRCLE)	834
0407	16	0210	0115			16	MAX		-IS STORAGE AREA FULL)	480

0408	11	0261	0265		X	11	0		BRING BACK A SET)	650
0409	00	0000	2000		K 20001	00	0000	2000		2664
0410	69	1124	8003		O OF	69	50000	8003		752
0411	60	0284	0389		C OK	60	XXXXA		MAKE SEVERAL VARIABLE ORDERS)	1058
0412	10	8001	0969			10	8001		RESTORE TO POSITIVE	1380
0413	16	0708	0463			16	C2		IS IT ALSO RIGHT END OF WORD	1096
0414	65	0167	0985		N EVN	65	XXXX1	BOTH	EVEN. WE SHOULD USE FIRST ADDEND	1898
0415	46	0178	0569			46	EXITY		MAKE DRUM AVAILABLE)	976
0416	60	0108	1013		N SYM	60	HSYMB		SYMBOLIC ADDRESS. BRING SYMBOL	1496
0417	69	0246	0349		N ZP	69	C1		LAST LINE AND END OF WORD. MODIFY)	1112
0418	69	1015	0661		N LODD1	69	PRE	SUBR7	SUB 11 FOUND EQU OF SYMB. RESERVE	2076
0419	60	0622	0627		R LODD2	60	87THX	SB10B	N.G. FIX TO OMIT PUNCHING	2078
0420	69	1669	1224		R LODD3	69	DRUMT		BLANK ADDRESS. IS DRUM FULL	2080
0421	20	0182	1285	0182	N LODD4	20	PDD06	PDD06	DRUM. STORE ADDRESS FOR PUNCHING	2086
0422	69	1030	0983		R LOOD5	69	1D30		NEW SYMBOL. WHERE ARE WE	2094
0423	00	0000	0001		R LODD6	00	ODD0	SETCC	800X ADDRESS. OUIT IMMEDIATELY	2102
0424	20	0182	1335	0182	R LOD07	20	PDD06	PDD06	CORE ADDRESS. STORE FOR PUNCHING	2104
0425	00	0000	0419		R LODD8	00	00D0	LD002	MISCELLANEOUS ADDRESS. OUIT AND OMIT	2110
0426	00	0908	0280		J	00	0908		TAGS AND EXIT FOR SUBROUTINE 13	1254
0427	20	1555	0900	1555	O 800	20	AD201	09D0	AD2D1 TO RESTORE FOR NEXT LINE OF TABLE	1134
0428	65	0000	0756		J	65	0000		BRING ADDENDS	1842
0429	10	0382	8003			10		8003	MAKE DRUM AVAILABLE)	970
0430	69	0154	1110			69	R00D4		CONTROL INFORMATION FOR PUNCHING)	1926
0431	69	0515	0533			69	110TH	RU	BRING UNITY TO INDICATE UNRESERVATION	1036
0432	65	0435	0439			65	LSYMB		SYMBOL-STORING ORDER)	1330
0433	96	0386	0288			96		L	D, WITH 1. 1, WITH L.	670
0434	71	0177	8003			71	P0001	8003	PUNCH CARD. PERFORM UPPER.	1594
0436	99	0140	0291			99		MISC	MULTIPLE BRANCH ACCORDING TO TAGS)	740
0437	35	0002	0543			35	0002		MAKE SEVERAL VARIABLE ORDERS)	1074
0438	20	1679	0282	1679		20	BLANB	BLANB	SPEC. SUBR. BRING BACK BLANB + ORCEB	870
0439	15	0392	0747			15	SS		SYMBOL-STORING ORDER)	1332
0440	24	0743	0296	0743		24	SW	SW	FORWARD. SET SWITCH OFF FOR 1ST PART	1636
0441	24	0179	0532	0179	N SUB13	24	EXITZ	EXITZ	STORE EXIT	1778
0442	65	0435	0489			65	LSYMB		LOCATION OF EQUIVALENT RELATIVE TO E1	1338
0443	44	0497	0398			44		OUT2	STORE PROCESSED CARD)	888
0444	30	0000	0717		J	30	ODD0		SHIFT MODIFIED WORD BACK INTO PLACE	1728
0445	16	0498	0754			16	JBRL		SUBTRACT INITIAL BRINGING ORDER	1532
0446	65	1354	1020			65	60D	SA1	RESET BRINGING ORDER, AND SEARCH MORE	1682
0448	20	0167	0620	0167	N OUT	20	XXXX1	LOOP	STORE MODIFIED BRINGING ORDER + REPEAT	934
0449	46	0352	0791			46	XXXX	FULL	I, TABLE IS FULL	1526
0450	65	0197	0251			65	ALOPT		ALOFT TO LOWER	806
0451	10	0554	0509			10	P2	A	MAKE STORING ORDER)	1356
0452	10	0916	0980			10	0		DUPPLICATE INTO AVAILABILITY TABLE)	2414
0453	60	0206	0057			60		SB10A	PUNCH CARD	592
0454	69	0758	0561			69	C7		LAST LINE AND END OF WORD. MODIFY)	1116
0455	69	0908	0063			69	PRE	SUBR9	ANALYZE THE ADDRESS	1176
0456	60	0108	0263			60	SA		ARE WE TO LAST LINE OF TABLE)	1018
0457	45	0026	0111			45	OUITT		IF SO, OUIT SEARCH	486
0458	35	0001	0465			35	0001		REPLACE DIGIT OF AVAIL. WORD BY W)	1000
0459	20	0486	0335	0486	J	20	SAVOR	SB10C	STORE DYN LEV OF CORE, AND FINISH UP	1276
0460	65	0197	0520			65	ALOFT	X0003	MULTIPLE EXIT, ACCORDING TO KIND	694
0461	11	8001	0367			11	8001		MAKE SEVERAL VARIABLE ORDERS)	1066
0462	16	0038	0793			16	16THX	ST	COMPARISON CONSTANT FOR LAST PART	1642
0463	45	0216	0417			45		ZP	D, NO	1098
0464	15	0515	0514			15	110TH	UM	COUNT OF TAGS IN LOWER	1956
0465	65	8002	0723			65	8002		REPLACE DIGIT OF AVAIL. WORD BY W)	1002
0466	10	1119	0724			10	291XX		ADD 29	1478
0467	10	8001	1223		N RC	10	8001		CORE ADDRESS. RESTORE THE 2000	1394
0468	69	0197	0300		C NO	69	ALOFT		IS D AN ACTUAL ADDRESS	686
0469	20	1023	0476	1023	N D0001	20	BLANK	00D08	STORE EQUIV OF BLANK FOR REFERENCE	2124
0470	60	0672	0627		R D0002	60	88THX	SB10B	N.G. OUIT AND OMIT PUNCHING	1226
0471	69	0095	0748		R D0003	69	OPTIM		BLANK D. IS IT MOD OPERATION	2128
0472	69	1226	0677		N 00004	69		INDEX	DRUM ADDRESS. INDEX IF TAGGED	2136
0473	65	1276	0611		R DDD05	65		SUB11	NEW SYMBOL. FIND EQUIVALENT	2140
0474	69	0778	0441		R DDD06	69		SUB13	800X ADDRESS. FIND DYNAMIC LEVEL	2144
0475	69	0928	0677		R DDD07	69		INDEX	CORE ADDRESS. INDEX IF TAGGED	2152
0476	20	0184	1049	0184	R D0008	20	P0D08	B	MISC ADDRESS, USUALLY A CONSTANT	2164
0477	69	1065	0661		N 00009	69	PRE	SUBR7	STORE EQUIVALENT OF NEW SYMBOL	2170
0478	00	0000	0470		R 00010	00	0000	D0002	SYMB TABLE FULL OR DRUM PACKED. OUIT	2172
0479	21	0284	0387	0284		21	XXXXA	XXXXA	STORE INITIAL ADDRESS TO BE RESERVED	1040
0480	60	0151	0712			60	R00D1		ALPHARETICAL L TO UPPER	2070
0481	11	8001	0437			11	8001		MAKE SEVERAL VARIABLE ORDERS)	1072
0482	45	0636	0537			45	ABC		I, THE ADDRESS IS BLANK	1436
0483	69	0486	0339		N OVR	69	SAVOR		FINISHED. SAVE SAVAR AS ORCEO)	936
0484	69	0687	0390		N O1	69	F		PREPARE EXIT AND GO TO SUBROUTINE 13)	1618
0485	44	0416	0240			44	SYM		ARE LAST FOUR CHARACTERS DIGITS)	1458
0487	69	0190	0441			69		SUB13	GET DYNAMIC LEVEL OF CORE ADDRESS	1268
0488	24	0186	0285	0186	O O	24	P0D10	X	P0010	940
0489	14	0492	1029			14	20DXXX			1340
0490	65	0095	0749		N FIB0	65	OPTIM		LOCATION OF EQUIVALENT RELATIVE TO E1	1798
0491	15	0294	0335			15	XXXX2	SB10C	FORWARD I OR BACKWARD D. GET ADDENDS	1260
0492	00	0002	0000		K 2DXXX	00	0002	D000	ADD TO GIVE MODIFIED DYNAMIC LEFEL	2670
0493	44	0597	0448			44		OUT	BRING BACK A SET OF RESULTS)	930
0494	21	0186	1238	0186		21	PDD10	PROI	FIX C.I. SO THAT D WILL NOT PUNCH)	2282
0495	30	0006	0960			30	DD06		SHOULD WE INTERCHANGE ADDENDS)	1856
0496	65	1054	1020			65	799	SAI	START BACK FROM TOP OF TABLE	1696
0497	10	0650	8003			10	052	8003	STORE PROCESSED CARD)	890
0499	20	0186	0689	0186		20	P0D10		C.I. TO PUNCH X-9)	1584
0500	69	0753	0400			69		SUBR5	PROCESS D BACKWARDS	812
0501	15	8002	0909			15	8002		4 TIMES DYNAMIC ADDRESS IN D POSITION	1626
0502	10	1019	1232		J	10	DDIFF	ZO	ADD RLOCATION AMOUNT	2456

0503	90	0707	0358		90		A		IF NOT, JUMP AHEAD	606
0504	90	0335	0459		90	SB10C			D, WE SHOULD STORE THIS DYNAMIC LEVEL	1274
0505	69	0908	0661		69	PRE	SUBR7		USUAL CASE. STORE SYMBOL AND EQUIV	1214
0506	11	0409	0313		11	20001			IS INITIAL ADDRESS LESS THAN 2000	1044
0507	69	1030	0283		69	1030			WHERE DIO SEARCH START	536
0508	45	0162	0178		45		EXITY		O, WE HAVE NOT FINISHED. I, WE HAVE.	1012
0509	10	0294	0167		10	XXXX2	XXXX1		MAKE STORING ORDER)	1358
0510	65	0197	0527		65	ALOPT	X0010		MULTIPLE EXIT	718
0511	22	0359	0262	0359	22	SPR		SPR	MAKE SEVERAL VARIABLE ORDERS)	1090
0512	15	0765	1020		15	800	SAI		ADVANCE VARIABLE BRINGING ORDER	1654
0513	11	0266	0331		11	C6	SU		MODIFY TO TAKE FIRST LINE, SAME COLUMN	1110
0514	20	0545	0598	0545	20	COUNT		COUNT	STORE TAG-COUNT	1958
0515	00	0000	0001		00	0000	0001			2644
0516	65	0346	0651		65	AI			AFTER RESTART. BRING BRINGING ORDER	1684
0517	10	8001	1323		10	8001			WRONG SYMBOL. RESTORE UPPER TO PLUS	1428
0518	15	0075	0329		15	13RD	X1		ADDRESS N G. CALL UNFIXED	698
0519	69	1050	0653		69	1050			BLANK ADDRESS. IS THIS FIRST CARD	700
0520	00	0000	0329		00	0000	X1		ORUM, OLD SYMB, REG. FIXED	704
0521	15	0075	0329		15	13RD	X1		NEW SYMBOL. UNFIXED	706
0522	15	0075	0329		15	13RD	X1		800X. CALL IT UNFIXED	696
0523	00	0000	0519		00	0000	X0002		CORE. TREAT SAME AS BLANK	708
0524	00	0000	0329		00	0000	X1		OTHER ADDRESS. FIXED	710
0525	15	0132	0187		15	12NOX	X2		O IS N G. CALL UNFIXED	722
0526	69	1050	0703		69	1050			BLANK. IS THIS FIRST CARD	724
0527	00	0000	0081		00	0000	F		OLD SYMBOL, ORUM, REGION. FIXED	728
0528	15	0132	0187		15	12NOX	X2		NEW SYMBOL. UNFIXED	730
0529	15	0132	0187		15	12NOX	X2		800X ADDRESS. UNFIXED	720
0530	91	0526	0525		91	X0009	X0008		CORE. BRANCH, LIKE BLANK, ARITH, UNF	732
0531	01	0666	0531		01	0666	X0014		OTHER. ERROR. SHOULD NOT BE HERE.	734
0532	95	0735	0787		95		9		O, ADDRESS. I, FORWARD I OR BACKW L	1780
0533	24	0326	0479	0326	24	W		W	STORE O OR I	1038
0534	69	0737	0440		69	OFF			FORWARD. SET SWITCH OFF FOR 1ST PART	1634
0535	30	0001	0641		30	0001			THEY ARE DIGITS. GET VALUE)	1466
0537	61	0515	1069		61	110TH	00N		MODIFY EXIT AND PREPARE TO QUIT	1438
0538	00	0909	0342		00	0909			TAGS AND EXIT FOR USE IN SUBR 13	1204
0539	46	0542	0977		46		8A0		I, RELOCATED CORE ADDRESS TOO HIGH	1400
0540	04	0000	0000		04	0000	0000			1918
0541	35	0004	0451		35	0004			SHIFT TO 0 POSITION	1354
0542	10	8001	0713		10	8001	RES		RESTORE THE 9060	1402
0543	10	8003	0401		10	8003			MAKE SEVERAL VARIABLE ORDERS)	1076
0544	21	0186	1338	0186	21	P0010	ALL	P0010	FIX C.I. SO THAT I WILL NOT PUNCH)	2302
0546	67	8003	0178		67	8003	EXITY		IF NOT 8001. CORRECTION IS ZERO	2020
0547	15	0038	8002		15	16THX	8002		BRING BACK A SET OF RESULTS)	922
0548	93	1101	0785		93		SEO		O, OPERATION 31. I, 30,35,36.	1828
0549	60	1356	1088		60	A0002	TA			1766
0550	69	0953	0142		69		SUBR6		PROCESS I BACKWARDS	830
0551	16	1054	1009		16	799			SUBTRACT END OF TABLE	1650
0552	15	1266	0784		15	ST			MAKE STORING ORDER	2488
0554	22	1555	0179		22	E0001	EXITZ	E0001	INITIAL OF ORDER TO STORE 0-POSITION	1366
0555	20	0167	0670	0167	20	XXXX1	XXXX1		MAKE BRINGING ORDER)	1348
0556	00	0989	0268		00	0989	M0001		TAGS AND EXIT FOR SUBROUTINE 11	1190
0557	10	0260	0215		10	BRNG			NOT 1ST CARD. MAKE COMP. CONST.)	546
0558	46	0411	0212		46	OK			I, LAST ONE WOULD BE OVER 1999	1050
0559	46	0412	0613		46		FXT		I, FIRST IS NON-BLANK. FIXED ADDRESS.	1378
0560	65	0163	0113		65		SSB		BRING BACK BLANK AND ALOPT ONCE MORE	850
0561	24	0359	0362	0359	24	SPR		SPR	LAST LINE AND END OF WORD. MODIFY)	1118
0562	15	0038	0793		15	16THX	ST		COMPARISON CONSTANT FOR LAST PART	1666
0563	11	0316	0331		11	C8	SU		LAST LINE AND END OF WORD. MODIFY)	1122
0564	65	0902	0662		65	0902			BRING APPROPRIATE TAG	2044
0565	20	1355	0900	1355	20	A0001	0900	A0001	INITIAL OF STORING ORDER	1128
0566	22	0183	0986	0183	22	P0007	P0007	P0007	STORE OPERATION FOR PUNCHING	1934
0567	11	8003	0725		11	8003			ARE LAST FOUR CHARACTERS DIGITS)	1452
0568	00	0001	0375		00	0001	J0008		ADDEND AND EXIT FOR SUBROUTINE 18	1312
0569	10	0322	8003		10	OSB	8003		MAKE DRUM AVAILABLE)	978
0570	65	0553	0243		65	VAR	LOOP		BRINGING ORDER. BACK FOR ANOTHER CARD	894
0571	69	0074	0377		69	BRNG			MAKE NEW BRINGING ORDER)	906
0572	20	1023	0577	1023	20	BLANK	I0006	BLANK	STORE EQUIVALENT OF BLANK FOR REFERENC	2186
0573	60	0722	0627		60	89THX	SB108		N.G. QUIT AND OMIT PUNCHING	2188
0574	60	0152	1012		60	R0002			BLANK I. IS 0 ALSO BLANK	2190
0575	69	1028	0677		69		INOEX		ORUM ADDRESS. INDEX IF TAGGED	2204
0576	65	0979	0611		65		SUB11		NEW SYMBOL. FIND BEST VALUE.	2208
0577	20	0185	0177	0185	20	P0009	EXITX	P0009	800X ADDRESS. STORE AND EXIT	2212
0578	69	1331	0677		69		INDEX		CORE ADDRESS. INDEX IF TAGGED	2214
0579	20	0185	0177	0185	20	P0009	EXITX	P0009	OTHER ADDRESS. STORE AND EXIT.	2226
0580	20	0185	0988	0185	20	P0009		P0009	STORE EQUIVALENT OF NEW SYMBOL	2228
0581	00	0000	0573		00	0000	I0002		SYMBOL TABLE FULL. OMIT PUNCHING ADOR	2232
0582	97	0935	0937		97	P.			O, FORWARD. I, BACKWARD.	1852
0583	35	0004	0593		35	0004			MAKE SEVERAL VARIABLE ORDERS)	1082
0584	98	8009	0000		98	8009	0000			1986
0585	69	0038	0741		69	16THX			PRESET T AS POSITIVE 1)	1502
0586	24	0178	1231	0178	24	EXITY		EXITY	STORE EXIT INSTRUCTION	1992
0587	65	0290	8001		65	Y	8001		BRINGING ORDER INTO LOWER	1506
0588	46	0791	0642		46	FULL			O, TABLE IS FULL	1522
0589	10	0294	0399		10	XXXX2			FIRST CHARACTER INTO UPPER	1470
0590	65	0800	1306		65	0800			BRING OPTIMIZING ADDENDS AND TAGS	1964
0591	10	0394	0509	J	10	P3	A		MAKE STORING ORDER	1362
0592	65	1660	0665	J	65	G0001			BRING EQUIVALENT OF REGION	1486
0593	69	0246	0249		69	C1			MAKE SEVERAL VARIABLE ORDERS)	1084
0594	60	0095	1099	J	60	E			BRING SYMBOL-EQUIVALENT TO UPPER	2356
0595	65	0182	1337		65	P0006	MW		BRING BACK NUMERICAL ADDRESS	2092
0596	65	0158	0646		65	R0008	SDD		BRING NUMERICAL 0	2624

0597	10	0700	0547	10	QS	A	BRING BACK A SET OF RESULTS)	932
0598	60	0183	1237	60	P0007		OPERATION IN D OF UPPER	1960
0599	60	1357	1088	60	A0003	TA		1768
0600	24	0391	8002	24	ORCEB	8002	ORCEB SPEC. SUBR. BRING BACK BLANB + ORCEB	874
0601	16	1354	1059	16	600		SUBTRACT COMPARISON CONSTANT	1672
0602	20	0184	1340	20	P0008	P0008	STORE NEXT 4 DIGITS IN D POSITION	2540
0603	69	0306	0045	69		SUB20	PROCES D FORWARDS	766
0604	16	1208	0913	16	909		SUBTRACT 9090908995	1448
0605	35	0006	0970	35	0006		SHIFT TO REMOVE TABULAR ADDRESS)	1570
0606	16	0290	0345	16	Y		SYMB NOT IN TABLE. SUBTRACT Y	1518
0607	20	0155	0204	20	R0005	X	R0005 COMPARISON CONSTANT FOR BRINGING SET	596
0608	44	1316	0001	44	SETCC		I, IT IS ZERO AND WE HAVE FINISHED	2502
0609	16	1218	8002	16	KA	8002	RESTORE LOWER, AND A00 NEXT SYMBOL	2512
0610	20	1356	0000	20	A0002	0000	A0002 COMPARISON CONSTANT	984
0611	69	1669	1072	69	DRUMT		IS THE DRUM FULL	1608
0612	15	0915	1020	15	599	SAI	RESTORE AND MODIFY	1676
0613	65	8002	0177	65	8002	EXITX	FIXED ADDRESS. CLEAR UPPER AND EXIT.	1410
0614	10	0152	1312	10	R0002		ALPHARETIC O	2266
0615	69	0568	0677	69		INDEX	TO SUBROUTINE 18 TO INDEX	1310
0616	69	1669	1274	69	DRUMT		BLANK D AND I. CHECK ORUM TAG	2194
0617	24	1669	1122	24	ORUMT		CHANGE DRUM TAG TO 9)	1702
0618	46	0517	0972	46	NZ	DEF	D, IT WAS NOT RIGHT ONE . I, IT AS.	1426
0619	20	1003	0456	20	N		REDUCE NUMBER YET TO BE DONE)	1016
0620	60	0673	0057	60		SB10A	TO SUB 10A TO PUNCH ONE CARD	910
0621	16	0174	0379	16	QL		BRING BACK A SET OF RESULTS)	914
0622	00	0000	8000	00	0000	8000		2666
0623	45	0226	0160	45		R0010	IF SO, MULTIBRANCH + EXIT FROM QUITT	570
0624	00	0000	0376	00	0000	A	TAG-IDENTIFICATION, AND EXIT FROM 18	1246
0625	00	0988	0926	00	0988	C0009	EXIT AND TAGS FOR SUBROUTINE 11	1250
0626	65	1679	0371	65	BLANB	J0004	BRING EQUIVALENT OF BLANK BACKWARD L	1306
0627	10	0186	0941	10	P0010		FROM 4-6,19-21. TO SUPPRESS)	1596
0628	20	0903	1106	20	0903		0903 STORE I TAG	1946
0629	21	0294	0947	21	XXXX2		XXXX2 ALSO STORE FIRST LETTER OF ADDRESS)	1444
0630	60	0152	0762	60	R0002		ALPHARETICAL D INTO UPPER	2118
0631	20	0184	0487	20	P0008	P0008	STORE CORE ADDRESS FOR PUNCHING	1266
0632	95	1085	1087	95	CP		IF BACKWARD L, SUBTRACT TAG-COUNT)	1908
0633	65	0536	0220	65	EQUIV	F0003	BRING EQUIV AND MULTIBRANCH	1152
0634	11	8001	8002	11	8001	8002	DUPLICATE INTO AVAILABILITY TABLE)	2418
0635	15	1022	1027	15	21XXX		ADVANCE CARD NUMBER)	1590
0636	21	0108	0961	21	HSYMB		ADDRES NOT BLANK. STORE SYMBOL	1440
0637	65	0340	8001	65		8001	BRINGING ORDER INTO LOWER	1514
0638	14	0492	0904	14	20XXX		DIVIDE BY 2	1536
0639	10	0592	8003	10		8003	MAKE BRINGING ORDER)	1484
0640	65	1043	1347	65	8001		8001 INTO RIGHT END LOWER	2012
0641	20	0536	0589	20	EQUIV		STORE THESE 4 DIGITS	1468
0642	16	0395	0449	16	ZTABL			1524
0643	20	0384	0637	20	T		SHORT. PRESET T AS -1)	1512
0644	61	1248	0966	61	SEND		DUPLICATE INTO AVAILABILITY TABLE)	2420
0645	28	0486	8002	28	SAVOR	8002	SAVOR STORE ORCEO FOR USE WHEN WE START FORW	2242
0646	20	1019	0696	20	DOIFF		DOIFF STORE DRUM RELOCATION AMOUNT	2626
0647	24	1669	0072	24	DRUMT		PRESET DRUM TAG)	966
0648	69	8003	1261	69	8003		CLEAR DISTRIBUTOR	1968
0649	19	0320	1074	19	501XX		WHAT CELL DID WE RESERVE)	1756
0650	69	0181	8002	69	P0005	8002	RESTORE CONSTANT AND MODIFY	898
0651	16	1097	0701	16	A0		COMPARISON CONSTANT, SUBTRACTED	1686
0652	20	0183	8003	20	P0007	8003	P0007 STORE LOCATION OF AVAILABILITY WORD	2572
0653	90	0329	0521	90	X1	X0004	FIRST, UNFIXED. OTHER, FIXED.	702
0654	11	1308	0963	11	99THX		NON-ZERO. SUBTRACT 90	1474
0655	15	0959	1020	15	BP1	SAI	RESTORE AND MODIFY	1690
0656	93	0660	1061	93	XAS		D, IT IS AN INDEXING OPERATION	1810
0657	69	0310	0042	69		SUB19	STARTED WITH L, OR NOT FIRST. PROC. L	582
0658	44	1063	0314	44		LH	D, ADDR IS ODD. EQUIV IN RIGHT END	1544
0659	60	0184	0939	60	P0008		INDEXING OP, FORWARD I, BACKWARD L.	1832
0660	24	0179	0432	24	EXITZ		STORE EXIT	1328
0661	15	8002	1221	15	8002		DOUBLF IT	2046
0662	46	0366	0467	46		RC	D, DRUM. I, CORE.	1386
0663	10	0153	0714	10	R0003		ALPHARETIC I	2286
0664	45	0668	1219	45		NG	I, REGION IS UNDEFINED	1488
0665	44	1326	1078	44	NGD		D, RELOCATED 0 IS EXCESSIVE	2270
0666	11	8003	0975	11	8003		PUT INTO DISTRIBUTOR	1720
0667	15	0536	0691	15	EQUIV		GET EQUIVALENT OF REGIONAL ADDRESS)	1490
0668	16	0515	0719	16	110TH		MODIFY TO END WITH 199)	1054
0669	44	1073	0674	44	IPOS		O, USE I-POSITION. I, USE D-POSITION.	1350
0670	46	0324	0225	46		TP	O, NOT LAST LINE.	1022
0671	00	0000	0800	00	0000	0800		2662
0672	65	0167	0621	65	XXXX1		BRING BACK A SET OF RESULTS)	912
0673	65	0536	0541	65	EQUIV		BRING EQUIVALENT	1352
0674	00	0990	0368	00	0990	J0001	TAGS AND EXIT FOR SUBROUTINE 11	1316
0675	10	0409	0713	10	20001	RES	RESTORE RELOCATED DRUM ADDRESS	1392
0676	24	0178	1281	24	EXITX		STORE EXIT	2032
0677	44	1081	0682	44	YES		I, D IS NOT TRUE ADDRESS	1974
0678	65	8001	0535	65	8001		THEY ARE DIGITS. GET VALUE)	1464
0679	60	0153	0962	60	R0003		ALPHARETIC I TO UPPER	2180
0680	11	0334	0539	11	9060		SUBTRACT 9060	1398
0681	60	1235	1089	60	8	AX	MORE TAGS INTO UPPER	1976
0682	66	8002	0491	66	8002		CHANGE SIGN OF CORRECTION	1258
0683	15	1119	0734	15	291XX		ADD 29	2482
0684	10	8001	1291	10	8001		AVAILABILITY WORD BACK INTO UPPER	1724
0685	66	0038	0643	66	16THX		SHORT. PRESET T AS -1)	1510
0686	00	0000	0991	00	0000		TO BUILD EXIT FROM SUBROUTINE 13	1622
0687	65	0095	0699	65	OPTIM		FORWARD O. L-O ADDENOS, LEFT ENO LOWE	1786

0689	60	8003	0057		60	8003	SB10A	CLEAR LOWER	1586
0690	69	0794	1025		69		SUB2R	RESERVE 0000	2430
0691	16	0515	1269		16	110TH		GET EQUIVALENT OF REGIONAL ADDRESS)	1492
0692	24	0743	0396	0743	24	SW	SW	GOING BACKWARD. SET SWITCH OFF)	1660
0693	69	0178	0981	C SSW	69	EXITY		ALL CASES. GOING WHICH DIRECTION	1630
0694	60	8001	0709		60	8001	SUB10	PUNCH A CARO, AND REPEAT RBO	2408
0695	00	0500	0500	K C3	00	0500	0500	TO MODIFY IDENTIFICATION	2610
0696	65	0153	0746		65	R0003		ALPHABETIC I	2628
0697	24	1050	1103	1050	24	1050	1050	CHANGE FIRST-CARO TAG)	1186
0698	66	0515	0178		66	110TH	EXITY	IF 8001, CORRECTION IS -1	2018
0699	35	0001	1055		35	0001		L-0 AOOENOS TO LEFT ENO LOWER)	1788
0700	24	0187	0285	0187	24	P0011	X	P0011	942
0701	45	0655	0705		45		FULL	I, THERE IS NO AVAILABLE LOCATION.	1688
0702	00	0001	0001	P C6	00	0001	0001	TO MODIFY VARIABLE BRINGING ORDER	2602
0703	90	0187	0528		90	X2	X0011	NO, FIXED. YES, UNFIXED.	726
0704	35	0006	0770	N ABS	35	0006		ABSOLUTE ADDRESS OR EQUIVALENT	1560
0705	01	0222	1109	N FULL	01	0222		STOP BECAUSE ORUM IS PACKED	1698
0706	69	0095	0548		69	OPTIM		OPTIMIZING TAGS	1826
0707	69	1030	0383		69	1030		IF SO, WHERE OIO SEARCH START	608
0708	30	0009	0108	P C2	30	0009	SA	PRESETTER AND ALSO COMPARISON CONSTANT	1126
0709	15	1044	0499	C SUB10	15	81STX		C.1. TO PUNCH X-9)	1582
0710	35	0002	0785		35	0002	SEO	OTHER. 1-AOOENOS TO LEFT ENO LOWER	1804
0711	65	0536	0921	J PRE	65	EOUIV	C0004	BRING EQUIVALENT. MULTIBRANCH	1228
0712	69	1015	0063		69	PRE	SUBR9	ANALYZE L-ADDRESS	2072
0713	21	0167	0720	0167	21	XXXX1	XXXX1	STORE ADDRESS TEMPORARILY	1404
0714	69	1117	1024		69		SUBR8	PROCESS I	2288
0715	15	0718	8002		15		8002	MAKE BRINGING ORDER	1540
0716	61	8003	0780		61	8003		CHANGE ITS SIGN	2396
0717	60	8003	1225		60	8003		CLEAR LOWER	1730
0718	69	1555	0659	J	69	E0001		BRING WORD SHOWING EQUIVALENT	1542
0719	20	1003	0411	1003	20	N	OK	MODIFY TO ENO WITH 1999)	1056
0720	69	1273	1025		69		SUB2R	RESERVE IF ORUM ADDRESS	1406
0721	30	0004	0319		30	0004	LR	EQUIVALENT IN I POSITION OF LOWER	1550
0722	00	0000	0080	K 89THX	00	0000	0080		2658
0723	10	0326	0281		10	W		REPLACE DIGIT OF AVAIL. WORD BY W)	1004
0724	46	0416	0328		46	SYM		O, FIRST IS SPEC. CHAR. SYMBOLIC	1480
0725	45	0278	0679		45		OUT	ARE LAST FOUR CHARACTERS DIGITS)	1454
0727	00	0000	0631	J	00	0000		AOOEND AND EXIT FOR SUBROUTINE 18	1264
0728	65	1023	0421		65	BLANK	L0004	USE PROPER VALUE FOR BLANK ADDRESS	2084
0729	84	1200	0775		84	1200		SEARCH TABLE ACCORDING TO SIZE OF AOR	1564
0730	21	0158	1314	0158	21	R0008	R0008	STORE ADVANCED O ADDRESS	2392
0731	15	0384	8002		15	T	8002	ADVANCE LOWER AND BRING ANOTHER	1432
0732	10	8001	0999	N ORM	10	8001	A	ORUM. MAKE POSITIVE AGAIN	2058
0734	46	0029	0940		46	0029		IF SPEC CHAR. TO BLR DIRECTLY	2484
0735	97	0688	0490		97	F0	FIBO	O, FORWARD O. 1, BACKWARD O.	1782
0736	65	8003	0693		65	8003	SSW	CLEAR ACCUMULATOR AND JUMP	1616
0737	65	0346	0551	J OFF	65	AI	SWOF	HAVE WE REACHED TOP OF TABLE	1648
0738	65	1342	0611		65		SUB11	FIND AN EQUIVALENT	2098
0739	65	0346	0601	J OFB	65	AI		START BACKWARDS SEARCH. HOW FAR	1670
0740	24	0179	1282	0179	24	EXITZ	EXITZ	STORE EXIT	2458
0741	24	0384	0587	0384	24	T	T	PRESET T AS POSITIVE 1)	1504
0742	65	0197	1051		65	ALOFT		BACKWARD L. GET ALOFT	1806
0743	65	0346	0551	N BL	65	AI	SWOF	NOT PERFORMED—JUST FOR OPTIMIZING	1646
0744	11	8001	0452	SW	11	8001		DUPLICATE INTO AVAILABILITY TABLE)	2412
0745	15	0695	1349		15	C3		MODIFY IDENTIFICATION OF WORD	2566
0746	45	0796	0946		45		SCO	JUMP IF I IS BLANK	2630
0747	69	0108	8002		69	HSYMB	8002	STORE SYMBOL)	1334
0748	94	0800	0912		94	0800		IF BLANK O AND MOF, STOP MACHINE	2130
0749	92	0402	1105		92	SHOP		O, IT IS A SHIFT OPERATION	1800
0750	21	0304	0458	0304	21	UH	UH	REPLACE DIGIT OF AVAIL. WORD BY W)	998
0751	69	0955	1209		69	SR		NEW AVAILABILITY WORD INTO ORUM)	1734
0752	10	1245	0952		10	OS		RESTORE VARIABLE BRINGING ORDER	2596
0753	69	0356	0305	J	69		SUBR4	PROCESS L BACKWARDS FROM O	814
0754	20	0435	0638		20	LSYMB	LSYMB	ADDRESS OF SYMBOL RELATIVE TO START	1534
0755	65	0346	0951	C SKP	65	AI		ALL CASES. WHAT OIO WE RESERVE)	1740
0756	35	0004	0767		35	0004		SHIFT AOOENOS TO LEFT ENO	1844
0757	69	0360	0013		69	BRNG		BRING BACK A SET)	636
0758	30	0000	0108	P C7	30	0000	SA	PRESETTER FOR START OF NEW COLUMN	1140
0759	21	0186	0177	0186	21	P0010	EXITX	FROM 4-6,19-21. TO SUPPRESS)	1600
0760	30	0005	0974		30	0005		ISOLATE LAST DIGIT OF O-ADDRESS)	1820
0761	00	0988	0918	J	00	0988	C0001	TAGS AND EXIT FOR SUBROUTINE 11	1242
0762	69	1065	0063		69	PRE	SUBR9	ANALYZE O-ADDRESS	2120
0763	44	0517	0618		44	NZ		O, IT WAS NOT RIGHT ONE	1424
0764	69	1217	0063	C BOTH	69		SUBR9	ANALYZE I	2316
0765	60	1555	1088	Q 800	60	A0201	TA		1772
0766	16	0930	1339		16	94I		IS THIS ANOTHER RBO CARO)	2404
0767	92	0785	1222		92	SEO		JUMP UNLESS OPERATION 80,82, OR 88	1846
0768	00	0889	0572	J	00	0889	I0001	TAGS AND EXIT FOR SUBROUTINE 11	2202
0769	30	0010	0108	O C5	30	0010	SA	TO RESTORE AFTER SUBTRACTING C2	1136
0770	16	8002	0729		16	8002		ADDRESS INTO LEFT ENO DISTRIBUTOR	1562
0771	35	0002	1327		35	0002		OOO. SEPARATE LAST AOOENO.	1894
0772	90	0626	0369		90		J0002	O, ORUM IS NOT FULL	1304
0773	21	0228	0481	0228	21	P	P	MAKE SEVERAL VARIABLE ORDERS)	1070
0774	00	0000	0004	K 41XXX	00	0000	0004		2648
0775	15	0378	8002		15		8002	MAKE BRINGING ORDER	1566
0776	23	0536	0704	0536	23	EOUIV	ABS	STORE EQUIVALENT OF SYMBOL	1554
0777	11	0409	0663		11	20001		SUBTRACT 2000	1384
0778	00	0808	0782	J	00	0808		TAGS AND EXIT FOR USE IN SUBR 11	2146
0779	20	0738	0936	0733	20	B	B	SHOULD WE INTERCHANGE AOOENOS)	1860
0780	46	1329	0094		46	OUT	0094	O, WE HAVE FINISHED	2398
0781	44	0585	0686		44		SHRT	O, LONG SYMBOL	1500

0782	69	1086	0586		69	SUB17	PERHAPS MODIFY DYNAMIC LEVEL)	2148
0783	30	0004	1093		30	0004	0-POSITION HOLDS 0 FOR 0, 1 FOR 1)	2038
0784	69	0158	8002		69	R0008	ADDRESS OF 0001 OF REGION	2490
0785	69	0179	0582		69	EXITZ	ALL CASES. WHICH DIRECTION	1850
0786	69	0739	0692		69	OFB	GOING BACKWARD. SET SWITCH OFF)	1658
0787	97	0490	0742		97	FIBO	0, FORWARD I. I, BACKWARD L.	1784
0788	69	1243	0441		69		FIND DYNAMIC LEVEL	2158
0789	35	0009	0760		35	0009	ISOLATE LAST DIGIT OF 0-ADDRESS)	1818
0790	16	1308	1216		16	99THX	SUBTRACT 90	2478
0791	01	0111	1219		01	0111	STOP IF SYMBOL TABLE IS FULL	1556
0793	20	1097	1020	1097	20	A0	STORE COMPARISON CONSTANT	1668
0794	60	0160	0709		60	R0010	PUNCH CARD AND PROCEED	2432
0795	15	0695	0652		15	C3	MODIFY IDENTIFICATION OF WORD	2570
0796	65	0159	0946		65	R0009	BRING NUMERICAL I	2632
0797	15	1000	0555		15	P1	MAKE BRINGING ORDER)	1346
0798	20	0292	1049	0292	20	ORCEQ	STORE DYNAMIC LEVEL FOR REFERENCE	2162
0799	16	8002	0910		16	8002	0-ADDRESS TO LEFT END OF DISTR)	1836
0800	00	0404	9998	K	0800	00	FOR OPERATION = LOCATION - 0800	110
0801	00	0404	9998	K	0801	00	FOR OPERATION = LOCATION - 0800	112
0802	33	2322	8999	K	0802	33	FOR OPERATION = LOCATION - 0800	114
0803	00	0505	8998	K	0803	00	FOR OPERATION = LOCATION - 0800	116
0804	00	0505	8998	K	0804	00	FOR OPERATION = LOCATION - 0800	118
0805	00	0505	8998	K	0805	00	FOR OPERATION = LOCATION - 0800	120
0806	00	0505	8998	K	0806	00	FOR OPERATION = LOCATION - 0800	122
0807	00	0505	8998	K	0807	00	FOR OPERATION = LOCATION - 0800	124
0808	33	1212	9999	K	0808	33	FOR OPERATION = LOCATION - 0800	126
0809	33	0202	9999	K	0809	33	FOR OPERATION = LOCATION - 0800	128
0810	33	0504	9999	K	0810	33	FOR OPERATION = LOCATION - 0800	130
0811	33	0504	9999	K	0811	33	FOR OPERATION = LOCATION - 0800	132
0812	99	9999	8999	K	0812	99	FOR OPERATION = LOCATION - 0800	134
0813	99	9999	8999	K	0813	99	FOR OPERATION = LOCATION - 0800	136
0814	33	1110	8999	K	0814	33	FOR OPERATION = LOCATION - 0800	138
0815	33	0504	9999	K	0815	33	FOR OPERATION = LOCATION - 0800	140
0816	33	0504	9999	K	0816	33	FOR OPERATION = LOCATION - 0800	142
0817	33	0504	9999	K	0817	33	FOR OPERATION = LOCATION - 0800	144
0818	33	0504	9999	K	0818	33	FOR OPERATION = LOCATION - 0800	146
0819	33	2120	8999	K	0819	33	FOR OPERATION = LOCATION - 0800	148
0820	54	0303	9999	K	0820	54	FOR OPERATION = LOCATION - 0800	150
0821	45	0303	9999	K	0821	45	FOR OPERATION = LOCATION - 0800	152
0822	34	0303	9999	K	0822	34	FOR OPERATION = LOCATION - 0800	154
0823	34	0303	9999	K	0823	34	FOR OPERATION = LOCATION - 0800	156
0824	33	0303	9999	K	0824	33	FOR OPERATION = LOCATION - 0800	158
0825	44	0505	9999	K	0825	44	FOR OPERATION = LOCATION - 0800	160
0826	00	0505	9998	K	0826	00	FOR OPERATION = LOCATION - 0800	162
0827	00	0505	9998	K	0827	00	FOR OPERATION = LOCATION - 0800	164
0828	33	1212	9999	K	0828	33	FOR OPERATION = LOCATION - 0800	166
0829	33	0202	9999	K	0829	33	FOR OPERATION = LOCATION - 0800	168
0830	00	0000	9988	K	0830	00	FOR OPERATION = LOCATION - 0800	170
0831	00	0000	9988	K	0831	00	FOR OPERATION = LOCATION - 0800	172
0832	33	2726	8999	K	0832	33	FOR OPERATION = LOCATION - 0800	174
0833	33	2726	8999	K	0833	33	FOR OPERATION = LOCATION - 0800	176
0834	33	2726	8999	K	0834	33	FOR OPERATION = LOCATION - 0800	178
0835	00	0000	9988	K	0835	00	FOR OPERATION = LOCATION - 0800	180
0836	00	0000	9988	K	0836	00	FOR OPERATION = LOCATION - 0800	182
0837	33	2726	8999	K	0837	33	FOR OPERATION = LOCATION - 0800	184
0838	33	2726	8999	K	0838	33	FOR OPERATION = LOCATION - 0800	186
0839	33	2726	8999	K	0839	33	FOR OPERATION = LOCATION - 0800	188
0840	33	0404	9998	K	0840	33	FOR OPERATION = LOCATION - 0800	190
0841	33	0404	9998	K	0841	33	FOR OPERATION = LOCATION - 0800	192
0842	33	0404	9998	K	0842	33	FOR OPERATION = LOCATION - 0800	194
0843	33	0404	9998	K	0843	33	FOR OPERATION = LOCATION - 0800	196
0844	34	0405	9998	K	0844	34	FOR OPERATION = LOCATION - 0800	198
0845	43	0504	9998	K	0845	43	FOR OPERATION = LOCATION - 0800	200
0846	33	0404	9998	K	0846	33	FOR OPERATION = LOCATION - 0800	202
0847	33	0505	9998	K	0847	33	FOR OPERATION = LOCATION - 0800	204
0848	33	0404	9998	K	0848	33	FOR OPERATION = LOCATION - 0800	206
0849	33	0404	9998	K	0849	33	FOR OPERATION = LOCATION - 0800	208
0850	00	0000	9898	K	0850	00	FOR OPERATION = LOCATION - 0800	210
0851	00	0000	9898	K	0851	00	FOR OPERATION = LOCATION - 0800	212
0852	00	0000	9898	K	0852	00	FOR OPERATION = LOCATION - 0800	214
0853	00	0000	9898	K	0853	00	FOR OPERATION = LOCATION - 0800	216
0854	44	0505	9999	K	0854	44	FOR OPERATION = LOCATION - 0800	218
0855	00	0505	8998	K	0855	00	FOR OPERATION = LOCATION - 0800	220
0856	00	0505	8998	K	0856	00	FOR OPERATION = LOCATION - 0800	222
0857	00	0505	8998	K	0857	00	FOR OPERATION = LOCATION - 0800	224
0858	00	0000	9898	K	0858	00	FOR OPERATION = LOCATION - 0800	226
0859	00	0000	9898	K	0859	00	FOR OPERATION = LOCATION - 0800	228
0860	33	0504	9999	K	0860	33	FOR OPERATION = LOCATION - 0800	230
0861	33	0504	9999	K	0861	33	FOR OPERATION = LOCATION - 0800	232
0862	99	9999	8999	K	0862	99	FOR OPERATION = LOCATION - 0800	234
0863	99	9999	8999	K	0863	99	FOR OPERATION = LOCATION - 0800	236
0864	33	1110	8999	K	0864	33	FOR OPERATION = LOCATION - 0800	238
0865	33	0504	9999	K	0865	33	FOR OPERATION = LOCATION - 0800	240
0866	33	0504	9999	K	0866	33	FOR OPERATION = LOCATION - 0800	242
0867	33	0504	9999	K	0867	33	FOR OPERATION = LOCATION - 0800	244
0868	33	0504	9999	K	0868	33	FOR OPERATION = LOCATION - 0800	246
0869	33	0303	9999	K	0869	33	FOR OPERATION = LOCATION - 0800	248
0870	00	0504	8999	K	0870	00	FOR OPERATION = LOCATION - 0800	250
0871	00	0504	8999	K	0871	00	FOR OPERATION = LOCATION - 0800	252
0872	00	0504	8999	K	0872	00	FOR OPERATION = LOCATION - 0800	254
0873	00	0504	8999	K	0873	00	FOR OPERATION = LOCATION - 0800	256

0874	00	0504	8999	K	0874	00	0504	8999	FOR OPERATION = LOCATION - 0800	258
0875	00	0504	8999	K	0875	00	0504	8999	FOR OPERATION = LOCATION - 0800	260
0876	00	0504	8999	K	0876	00	0504	8999	FOR OPERATION = LOCATION - 0800	262
0877	00	0504	8999	K	0877	00	0504	8999	FOR OPERATION = LOCATION - 0800	264
0878	00	0504	8999	K	0878	00	0504	8999	FOR OPERATION = LOCATION - 0800	266
0879	00	0505	9999	K	0879	00	0505	9999	FOR OPERATION = LOCATION - 0800	268
0880	01	0100	9898	K	0880	01	0100	9898	FOR OPERATION = LOCATION - 0800	270
0881	00	0000	9898	K	0881	00	0000	9898	FOR OPERATION = LOCATION - 0800	272
0882	01	0100	9898	K	0882	01	0100	9898	FOR OPERATION = LOCATION - 0800	274
0883	00	0000	9898	K	0883	00	0000	9898	FOR OPERATION = LOCATION - 0800	276
0884	33	2526	8999	K	0884	33	2526	8999	FOR OPERATION = LOCATION - 0800	278
0885	00	0606	8998	K	0885	00	0606	8998	FOR OPERATION = LOCATION - 0800	280
0886	00	0606	8998	K	0886	00	0606	8998	FOR OPERATION = LOCATION - 0800	282
0887	00	0606	8998	K	0887	00	0606	8998	FOR OPERATION = LOCATION - 0800	284
0888	01	0100	9898	K	0888	01	0100	9898	FOR OPERATION = LOCATION - 0800	286
0889	00	0000	9898	K	0889	00	0000	9898	FOR OPERATION = LOCATION - 0800	288
0890	44	0505	9998	K	0890	44	0505	9998	FOR OPERATION = LOCATION - 0800	290
0891	33	0505	9998	K	0891	33	0505	9998	FOR OPERATION = LOCATION - 0800	292
0892	33	0505	9998	K	0892	33	0505	9998	FOR OPERATION = LOCATION - 0800	294
0893	33	0505	9998	K	0893	33	0505	9998	FOR OPERATION = LOCATION - 0800	296
0894	33	0505	9998	K	0894	33	0505	9998	FOR OPERATION = LOCATION - 0800	298
0895	33	0505	9998	K	0895	33	0505	9998	FOR OPERATION = LOCATION - 0800	300
0896	33	0505	9998	K	0896	33	0505	9998	FOR OPERATION = LOCATION - 0800	302
0897	33	0505	9998	K	0897	33	0505	9998	FOR OPERATION = LOCATION - 0800	304
0898	33	0505	9998	K	0898	33	0505	9998	FOR OPERATION = LOCATION - 0800	306
0899	44	0505	9998	K	0899	44	0505	9998	FOR OPERATION = LOCATION - 0800	308
0900	65	1003	0508	R	0900	65	N		HAVE WE FINISHED	1010
0904	35	0004	0715			35	0004		QUOTIENT INTO D OF LOWER	1538
0905	35	0001	1011			35	0001		RESERVE)	1716
0907	16	0410	0315			16	OF		IS THIS LAST SET)	664
0908	65	0536	0271	J PRE	65	EOUIV	M0004		EQUIVALENT TO LOWER. MULTIBRANCH	1178
0909	15	8002	0693		15	8002	SSW		4 TIMES DYNAMIC ADDRESS IN D POSITION	1628
0910	84	1300	1275		84	1300			LOOK UP IN TABLE	1838
0911	65	0536	0371	J PRE	65	EOUIV	J0004		BRING EQUIVALENT. MULTIBRANCH	1296
0912	65	1115	0611		65		SUB11		BLANK D, BUT NOT MOF. FIND VALUE OF O	2132
0913	46	0416	0567		46	SYM	LOO		D, ADDRESS IS SYMBOLIC.	1450
0914	69	1267	0063		69		SUBR9		ANALYZE D	2336
0915	60	1354	1088	O 599	60	A0000	TA			1762
0916	24	1555	0000	O O	24	A0201		0000	A0201	2434
0917	16	0733	0935		16	B	P		INTERCHANGE AND MAKE NEGATIVE)	1880
0918	20	1023	0376	1023 N C0001	20	BLANK	A	BLANK	STORE EQUIV OF BLANK FOR FORWARD L	1230
0919	60	0672	0627	R C0002	60	88THX	S810B		N G. OMIT PUNCHING	1232
0920	69	1050	1353	R C0003	69	1050			BLANK ADDRESS. WAS THERE BLANK BACK L	1234
0921	69	0624	0677	N C0004	69		INDEX		DRUM OR EQUIVALENT. INDEX IF TAGGED	1244
0922	65	0625	0611	R C0005	65		SUB11		NEW SYMBOL. FIND BEST EQUIVALENT	1248
0923	69	0426	0441	R C0006	69		SUB13		800X ADDRESS. GET DYNAMIC LEVEL	1252
0924	69	0727	0677	R C0007	69		INDEX		CORE ADDRESS. INDEX IT	1262
0925	20	0184	0177	R C0008	20	P0008	EXITX	P0008	MISC ADDR. STORE FOR PUNCHING. EXIT	1280
0926	69	0711	0661	N C0009	69	PRE	SUBR7		STORE EQUIVALENT OF NEW SYMBOL	1282
0927	00	0000	0919	R C0010	00	0000	C0002		NEW SYMBOL, BUT TABLE FULL. N.G.	1284
0928	00	0000	0932	J	00	0000			ADDEND AND EXIT FOR SUBROUTINE 18	2154
0929	46	0732	0933		46	DRM			I, CORE. D, DRUM.	2052
0930	00	0000	0094	K 94I	00	0000	0094			2436
0931	96	0484	0736		96	DI			I, ADDRESS IS BACKWARD I OR FORWARD L	1614
0932	20	0184	0788	0184	20	P0008		P0008	STORE THE CORE ADDRESS FOR PUNCHING	2156
0933	11	8003	1242		11	8003			CORE. CLEAR UPPER	2054
0934	15	1090	8002		15	K	8002		MAKE STORING ORDER, AND STORE ZERO	2510
0935	20	0167	1320	0167 C P	20	XXXX1		XXXX1	ALL CASES. STORE ADDENDS	1884
0936	60	8003	0943		60	8003			SHOULD WE INTERCHANGE ADDENDS)	1862
0937	20	0792	0495	0792	20	IND		IND	BACKWARD. SHOULD WE INTERCHANGE ADDENDS	1854
0938	69	1293	0441		69		SUB13		FIND DYNAMIC LEVEL	2220
0939	30	0004	0799		30	0004			D-ADDRESS TO LEFT END OF OISTR)	1834
0940	35	0004	0552		35	0004			SHIFT TO O POSITION	2486
0941	21	0154	0759	0154	21	R0004		R0004	TO SUPPRESS IN CASE WE SEARCH	1598
0942	16	0545	1085		16	COUNT	CP		IF BACKWARD L, SUBTRACT TAG-COUNT)	1912
0943	30	0002	0949		30	0002			SHOULD WE INTERCHANGE ADDENDS)	1864
0944	65	0152	1318	J	65	R0002			ALPHABETIC SYMBOL	2534
0945	10	0702	8003		10	C6	8003		MODIFY BRINGING ORDER, AND BRING	2576
0946	20	0726	0001	0726 C SCD	20	COIFF	SETCC	COIFF	STORE CORE RELOCATION AMOUNT	2634
0947	11	8001	0604		11	8001			CLEAR FIRST LETTER	1446
0948	91	0177	0335		91	EXITX	S810C		O, BRANCH. I, ARITHMETIC.	2168
0949	20	0906	1010	0906	20	A		A	SHOULD WE INTERCHANGE ADDENDS)	1866
0950	90	0271	0505		90	M0004			D, CASE WHERE WE DUPLICATE COMPUTATION	1212
0951	16	1354	1309		16	600			WHAT CELL DID WE RESERVE)	1742
0952	16	1295	1249		16	C7	LOOP		MODIFY IDENTIFICATION FOR NEXT CARD	2598
0953	69	0406	0305	J	69		SUBR4		PROCESS L BACKWARDS FROM I	832
0954	20	0959	0462	0959	20	BP1		BP1	STORE VARIABLE BRINGING ORDER	1640
0955	21	9972	0755	9972 J STR	21	9972	SKP	9972	NEW AVAILABILITY WORD INTO DRUM)	1738
0956	15	0062	1017		15	OPREG			ADD OLD DYNAMIC LEVEL	1902
0957	69	0460	0063		69		SUBR9		ANALYZE I, FOR TYPE OF ADDRESS	692
0958	69	0711	0063		69	PRE	SUBR9		ANALYZE THE ALPHABETIC D ADDRESS	1226
0960	35	0008	0779		35	0008			SHOULD WE INTERCHANGE ADDENDS)	1858
0961	30	0008	0629		30	0008			ALSO STORE FIRST LETTER OF ADDRESS)	1442
0962	69	1215	0063		69		SUBR9		ANALYZE I-ADDRESS	2182
0963	46	0466	0416		46		SYM		I, FIRST CHAR IS DIGIT. ADDR SYMBOLIC	1476
0964	16	0409	1064		16	20001			CORE OR DRUM	2364
0965	30	0005	0628		30	0005			I TAG TO RIGHT END LOWER	1944
0966	16	1230	8002		16		8002		DUPLICATE INTO AVAILABILITY TABLE)	2422
0967	14	1022	1060		14	21XXX			WHICH ONE DO WE USE)	1888
0968	15	0726	1082		15	CDIFF	CD		ADD CORE RELOCATION AMOUNT	2368
0969	60	8002	0777		60	8002			NUMERIC ADDRESS ALONE IN UPPER	1382

0970	30	0006	1069		30	0006	DDN	SHIFT TO REMOVE TABULAR ADDRESS)	1572		
0971	35	0001	0678		35	0001		ONE DIGIT INTO UPPER	1972		
0972	16	0290	0445	N DEF	16	Y		DEFINED SYMBOL. SUBTRACT Y	1530		
0973	69	0246	0299		69	C1		MODIFY TO TAKE NEXT COLUMN OF TABLE)	1102		
0974	10	1277	8003		10		8003	MAKE BRINGING ORDER	1822		
0975	35	0004	0685		35	0004		MOVE AMOUNT OF SHIFT INTO D	1722		
0976	20	0178	0931	0178	20	EXITX	EXITX	STORE EXIT OUT OF LOWER	1612		
0977	60	0622	0177	N BAD	60	87THX	EXITX	MODIFIED ADDR TOD HIGH. 8000 TO UPPER	1412		
0978	65	1023	0579		65	BLANK	10008	MAKE BLANK I EQUAL TO BLANK D	2198		
0979	00	0889	0580	J	00	0889	10009	TAGS AND EXIT FOR SUBR 11.	2210		
0980	46	1283	0634		46	DDNE		DUPLICATE INTO AVAILABILITY TABLE)	2416		
0981	97	0534	0786		97		BAK	D, FORWARD. I, BACKWARD.	1632		
0982	00	0000	1336	J	00	0000		ADDEND AND EXIT FOR INDEXING	2274		
0983	90	1001	0738		90	MOFLL		D, START SEARCH. I, WE ARE ON DUITT	2096		
0984	24	1030	1084	1030	24	1030	1030	TAG IN CASE L IS UNDEFINED SYMBOL)	2524		
0985	30	0008	0956	C BDTH	30	0008		BDTH. RIGHT ADDEND TO RIGHT END	1900		
0986	16	8001	0993		16	8001		CLEAR DP FROM ACCUMULATOR	1936		
0987	15	0132	1037		15	12NDX		SHOULD WE INTERCHANGE ADDENDS)	1870		
0988	69	0177	0661		69	EXITX	SUBR7	STORE SYMBOL AND ITS EQUIVALENT	2230		
0989	65	0155	1210		65	R0005		BRING TAGS AND OPERATION	1930		
0990	10	1898	0608	J	10	ZMAXM	U0001	ADD A SYMBOL TO UPPER	2500		
0991	35	0004	0501		35	0004		4 TIMES DYNAMIC ADDRESS IN D POSITION	1624		
0992	24	0184	1287	0184	24	P0008	P0008	STORE THE ADDRESS FOR PUNCHING	2006		
0993	35	0005	1006		35	0005		D TAG INTO UPPER	1938		
0994	60	0003	0057		60	READC	SB10A	PUNCH	2544		
0995	10	0702	8003		10	C6	8003	MODIFY BRINGING ORDER, AND BRING	2580		
0997	35	0001	0567		35	0001	LDD	ARE LAST FOUR CHARACTERS DIGITS)	1462		
0998	20	0292	0177	0292	20	ORCEO	ORCED	STORE DYNAMIC LEVEL FOR REFERENCE	2224		
0999	15	0167	0178	C A	15	XXXX1	EXITX	ADD ADDRESS BEING INDEXED. EXIT.	2060		
1000	69	1555	8003	P P1	69	E0001	8003	INITIAL OF BRINGING ORDER	1364		
1001	69	1004	0107	N MDFLL	69	L	A	STARTED WITH NEW-SYMBOL L	402		
1002	44	1005	0606	C LOOP	44		UND	I, SYMBOL IS NOT IN THE TABLE	1420		
1004	90	0090	0000	K L	90	0090	0000	TAG, SEARCH STARTED BY NEW-SYMBOL L	524		
1005	11	0108	0763		11	HSYMB		SOME SYMBOL WAS FOUND. SUBTRACT DURS	1422		
1006	21	0902	1056	0902	21	0902	0902	STORE D TAG	1940		
1007	65	0160	0165		65	R0010		IS IT TYPE 08)	494		
1008	65	1679	0376		65	BLANB	A	BRING EQUIVALENT OF BLANK BACKWARD L	1238		
1009	45	0512	1113		45		MAX	I, WE HAVE REACHED TOP OF TABLE	1652		
1010	15	0733	0987		15	B		SHOULD WE INTERCHANGE ADDENDS)	1868		
1011	30	0001	0667		30	0001		RESERVE)	1718		
1012	44	1265	0616		44	DN8		D, D IS NOT BLANK.	2192		
1013	35	0008	0781		35	0008		CLEAR ALL BUT LAST CHARACTER	1498		
1014	65	0167	0771		65	XXXX1		DDD. SEPARATE LAST ADDEND)	1892		
1015	65	0536	0421	J PRE	65	EQUIV	L0004	BRING EQUIVALENT. MULTIBRANCH	2074		
1016	45	1330	1332		45		BL	I, I IS UNPUNCHED	2462		
1017	15	0320	1325		15	501XX		ADD 50 TO MAKE SURELY POSITIVE	1904		
1018	60	0153	1214	J	60	R0003		IS THE I-ADDRESS BLANK	2384		
1020	20	0346	8001	0346	C SAI	20	AI	8001	AI	STORE BRINGING ORDER, AND DO IT	1656
1021	16	8002	1079		16	8002		WHAT CELL DID WE RESERVE)	1748		
1022	00	0000	0002	K 21XXX	00	0000	0002		2646		
1024	24	0177	0380	0177	N SUBR8	24	EXITX	EXITX	STORE EXIT	1374	
1025	24	0178	0381	0178	N SUB2R	24	EXITX	EXITX	ENTRY TO RESERVE. STORE EXIT.	1030	
1026	65	1229	1033		65	8003	EO	EVEN. PUT 8003 AT RIGHT END OF LOWER	2002		
1027	20	0181	0434	0181	N EVN	20	P0005	P0005	ADVANCE CARD NUMBER)	1592	
1028	00	0001	0577	J	00	0001	I0006	ADDENDS AND EXIT FOR USE IN SUBR 18.	2206		
1029	35	0004	1039		35	0004		LOCATION OF EQUIVALENT RELATIVE TO E1	1342		
1031	20	0536	0178	0536	J	20	EQUIV	EQUIV	STORE THIS ADDRESS IN EQUIV	1760	
1032	00	0001	1288		00	0001		ADDEND AND EXIT FOR INDEXING	2294		
1033	16	0536	0992	C ED	16	EQUIV		BDTH. SUBTRACT THE 800X ADDRESS	2004		
1034	16	0038	0744	A	16	16THX		DUPLICATE INTO AVAILABILITY TABLE)	2410		
1035	15	8001	1341		15	8001		WHAT CELL DID WE RESERVE)	1752		
1036	21	1091	1094	1091	21	DLO	OLD	SAVE DYNAMIC LEVEL OF L FROM 0)	822		
1037	14	0540	1048		14	42N0		SHOULD WE INTERCHANGE ADDENDS)	1872		
1038	16	1091	0245		16	OLD		WHICH L IS LESS, MEASURED ON CIRCLE)	840		
1039	20	0294	0797	0294	20	XXXX2	XXXX2	LOCATION OF EQUIVALENT RELATIVE TO E1	1344		
1040	30	0008	1118		30	0008		CLEAR ALL EXCEPT LAST CHARACTER)	2506		
1041	69	1044	1047		69	81STX		RESTORE FIRST CARD TAG)	366		
1042	65	0515	0178		65	110TH	EXITX	UNIT CORRECTION TO DYNAMIC LEVEL	2010		
1043	00	0000	8001	K 8001	00	0000	8001		2022		
1044	80	0000	0000	K 81STX	80	0000	0000		2676		
1045	10	0702	8003		10	C6	8003	MODIFY BRINGING ORDER, AND BRING	2584		
1046	44	0752	0003		44		READC	D, WE ARE NOT DONE.	2594		
1047	24	1050	0160	1050	24	1050	R0010	RESTORE FIRST CARD TAG) MULTIBRANCH	368		
1048	44	1351	1052		44	DK		I, WE SHOULD INTERCHANGE	1874		
1049	69	0095	0948	C B	69	DPTIM		SEVERAL CASES. WHAT KIND OF DP	2166		
1051	92	0402	0656		92	SHOP		D, IT IS A SHIFT OPERATION	1808		
1052	66	0906	1111		66	A		INTERCHANGE AND MAKE NEGATIVE)	1876		
1053	21	0658	0461	0658	21	X	X	MAKE SEVERAL VARIABLE ORDERS)	1064		
1054	60	1554	1088		60	A0200	TA		1770		
1055	16	8002	1313		16	8002		L-D ADDENDS TO LEFT END LOWER)	1790		
1056	65	8002	0965		65	8002		DELETE D TAG	1942		
1057	69	0510	0063		69		SUBR9	ANALYZE D	716		
1058	65	0761	0611	N BB1	65	SUB11		GET BLANK D DYNAMICALLY FROM I	1240		
1059	45	0612	1213		45	MIN		I, WE ARE NOT BACK TO START YET	1674		
1060	44	1014	0414		44	EVN		D, DPREG WAS DDD. I, EVEN.	1890		
1061	91	0364	0688		91	FD		D, IT IS A BRANCH OPERATION	1812		
1062	90	8002	1067		90	8002		D, WE ARE NOT ON FIRST BACKWARD CARD	2238		
1063	67	8001	1319		67	8001	LR	EQUIVALENT IS IN I POSITION OF LOWER	1546		
1064	46	1317	0968		46	D		D, DRUM. I, CORE.	2366		
1065	65	0536	0472	J PRE	65	EQUIV	00004	BRING EQUIVALENT. MULTIBRANCH	2122		
1066	16	0158	1332		16	R0008	BL	SUBTRACT NUMERICAL 0 FROM LOWER	2466		

1067	69	D292	0645		69	ORCEQ			STORE ORCEQ FOR USE WHEN WE START FORW	2240
1068	60	0158	1264		60	RDD08			BRING FIRST ADDRESS AGAIN	2388
1069	10	D179	80D3	C DON	10	EXITZ	8003		FROM MANY PLACES. MULTIBRANCH	1574
1070	24	0743	0446	D743	24	SW		SW	AT TOP. SET SWITCH ON)	1680
1071	21	D197	D177	D197	21	ALOPT		ALOPT	STORE ALL OPTIMIZING TAGS	1982
1072	90	0976	1077		90		EXITX		I, YES.	1610
1073	65	0536	0591	N IPOS	65	EQUIV			USE I-POSITION. BRING EQUIVALENT	1360
1074	15	8003	1031		15	8003			WHAT CELL DID WE RESERVE)	1758
1075	21	D178	0431	D178	24	EXITX		EXITX	ENTRY TO UNRESERVE. STORE EXIT.	1034
1076	65	1279	1033		65	8002	EO		OOO. PUT 8002 AT RIGHT END OF LOWER	2000
1077	15	0515	80D2	C PAKT	15	110TH	8D02		MODIFY EXIT)	1706
1078	69	0982	0677		69		INDEX		INOEX D	2272
1079	35	00D1	1035		35	DD01			WHAT CELL DID WE RESERVE)	1750
1080	65	80D3	1038		65	80D3			WHICH L IS LESS, MEASURED ON CIRCLE)	838
1081	60	0584	1089	N YES	60	9	AX		MORE TAGS INTO UPPER	1978
1082	15	0409	0124	C CD	15	2DD01	Q0007		RESTORE THE 200D AND JUMP BACK	2372
1083	10	0186	0494		10	P001D			FIX C.I. SO THAT O WILL NOT PUNCH)	2280
1084	24	0197	1350	D197	24	ALOPT		ALOPT	TAG IN CASE L IS UNDEFINED SYMBOL)	2526
1085	14	0320	1272	C CP	14	501XX			NEW LEVEL, MODULO 5D IN UPPER	1914
1086	15	0294	1049	J	15	XXXX2	B		PERHAPS MODIFY DYNAMIC LEVEL)	2150
1087	97	1085	0942		97	CP			IF BACKWARD L, SUBTRACT TAG-COUNT)	1910
1088	44	1241	0743	R TA	44	YES	SW		IS A LOCATION AVAILABLE IN THIS GROUP	1644
1089	10	0167	1071	C AX	10	XXXX1			ADD IN TAGS FROM OPTIM	1980
1090	11	0D00	0D01	P K	11	000D	D001		TO CHANGE ADDING ORDER TO STORING ONE	2514
1092	45	0546	0698		45	NZ			D, ADDRESS IS NOT 8001	2016
1093	35	0D04	1311		35	DD04			O-POSITION HOLDS 0 FOR D, 1 FOR I)	2040
1094	65	0197	0301		65	ALOPT			MODIFY TO USE ROUTINE FOR UNFIXED D)	824
1095	10	0702	8003		10	C6	80D3		MODIFY BRINGING ORDER, AND BRING	2588
1096	11	1352	1046		11	O			ARE WE DONE	2592
1098	60	0152	0914	C X	60	R00D2			ALPHABETIC D	2334
1099	69	00D1	1025		69	SETCC	SUB2R		RESERVE AND PUNCH CARO	2358
1100	90	0755	0905		90	SKP			O, SKIP BECAUSE COMP IS DOUBLE	1714
1101	35	0D04	0785		35	0004	SEO		OPERATION 31. TAKE LARGER ADDENOS.	1830
1102	65	0160	0766		65	R0D1D			IS THIS ANOTHER RBD CARD)	2402
1103	65	0556	0611		65		SUB11		FINO BEST EQUIVALENT TO BLANK	1188
1104	20	0959	0562	D959	20	BP1		BP1	INITIAL BRINGING ORDER)	1664
1105	93	0660	0710		93	XAS			O, IT IS AN INOEXING OPERATION	1802
1106	45	1260	1211		45		BT		O, I TAG IS NON-ZERO	1948
1107	46	0560	0311	C BOT	46		2ND		D, WE WILL USE ONE MEASURED FROM D	848
1108	69	0911	0D63		69	PRE	SUB9R		ANALYZE ALPHABETIC 1	1294
1109	69	0114	0617		69	915TX			CHANGE ORUM TAG TO 9)	1700
1110	21	0186	0989	0186	24	P0D1D		P001D	CONTROL INFORMATION FOR PUNCHING)	1928
1111	30	0D02	0917		30	DD02			INTERCHANGE AND MAKE NEGATIVE)	1878
1112	65	0157	1212	J	65	R0007			NUMERICAL L	2250
1113	69	0516	1070	N MAX	69	ON			AT TOP. SET SWITCH ON)	1678
1114	69	1018	1025		69		SUB2R		RESERVE FIRST CELL IN BANO	2382
1115	00	0888	0469	J	00	0888	00D01		TAGS AND EXIT FOR SUBROUTINE 11	2134
1116	30	0D08	0790		30	DD08			REGION DESIGNATOR TO RIGHT ENO)	2476
1117	44	1228	1278	J	44	NGI			O, RELOCATED I IS EXCESSIVE	2290
1118	44	0D01	0934		44	SETCC			O, OUIT BECAUSE WE HAVE FOUND LONG SYM	2508
1119	00	0D00	0D29		00	DD00	0D29	SW	SECOND PART, BACKWARDS. CHANGE SWITCH	2654
1120	24	0743	0496	0743	24	SW			O-POSITION HOLDS 0 FOR D, 1 FOR I)	1694
1121	65	0178	0783		65	EXITX			MODIFY EXIT)	2036
1122	65	0178	1077		65	EXITX	PAKT		I, RELOCATED ORUM WILL EXCEED 1999	1704
1123	46	0676	0977		46		BAC		FOR TELLING WHETHER DRUM OR CORE	1390
1124	00	0D0D	200D	K 27TH	00	0000	2D00		IF ORUM ADDRESS, USE REGULAR EXIT	2062
1200	19	9900	0D0D	K 1200	19	9900	0000		IF NOT AN ADDRESS, ADD 4	344
1201	79	9900	0D04	K 1201	79	9900	0004		IF 800X ADDRESS, ADD 2	346
1202	80	0300	0D02	K 1202	80	0300	DD02		IF NOT AN ADDRESS, ADD 4	348
1203	80	0400	0D04	K 1203	80	0400	0004		IF 800X ADDRESS, ADD 2	350
1204	80	0700	0D02	K 1204	80	0700	DD02		IF NOT AN ADDRESS, ADD 4	352
1205	89	9900	0D04	K 1205	89	9900	0004		IF NOT AN ADDRESS, ADD 4	354
1206	90	5900	0003	K 1206	90	5900	0D03		IF CORE ADDRESS, ADD 3	356
1207	99	9999	0D04	K 1207	99	9999	0004		IF NOT AN ADDRESS, ADD 4	358
1208	90	0900	8995	K 9D9	90	0900	8995			1576
1209	22	0167	80D1	0167	22	XXXX1	8001	XXXX1	NEW AVAILABILITY WORD INTO ORUM)	1736
1210	69	8003	0566		69	8003			CLEAR DISTRIBUTOR	1932
1211	10	0902	1310	C BT	10	0902			O TAG INTO UPPER	1952
1212	10	0151	1262		10	R0D01			ALPHABETIC L	2252
1213	69	0516	1120	N MIN	69	ON			SECOND PART, BACKWARDS. CHANGE SWITCH	1692
1214	45	1068	1329		45		OUT		O, NOT BLANK. I, IT IS BLANK.	2386
1215	65	0536	0575	J	65	EQUIV	I00D4		BRING EQUIVALENT. MULTIBRANCH	2184
1216	46	0684	0D29		46		D029		IF A OIGIT, TO BLR WITHOUT DESIGNATING	2480
1217	65	0536	0120	J	65	EQUIV	Q0D03		BRING EQUIVALENT. MULTIBRANCH.	2318
1218	11	0D01	0001	P KA	11	DD01	0001		TO CHANGE STORING ORDER TO ADDING ONE	2516
1219	61	1D22	1069	C NG	61	21XXX	DON		MODIFY EXIT FOR N G SYMBOL	1558
1220	14	0774	1021		14	41XXX			WHAT CELL DID WE RESERVE)	1746
1221	10	0167	1271		10	XXXX1			INOEXABLE ADDRESS INTO UPPER	2048
1222	16	0D95	0785		16	OPTIM	SEO		REDUCE ADDENOS BY 1 FOR 80,82, OR 88	1848
1223	10	0726	0681		10	CDIFF			ADD RELOCATION AMOUNT	1396
1224	90	0728	0419		90		L00D2		D, DRUM NOT FULL. I, ORUM IS FULL.	2082
1225	15	0346	0751		15	AI			NEW AVAILABILITY WORD INTO ORUM)	1732
1226	00	0000	0476	J	00	0000	D0DD8		ADDEND AND EXIT FOR USE IN SUBR 18	2138
1227	30	0D03	0785		30	DD03	SEO		L-O ADDENOS TO LEFT END LOWER)	1796
1228	30	0D02	1239	N NGI	30	D002			FIX C.I. SO THAT I WILL NOT PUNCH)	2298
1229	00	0D0D	80D3	K 80D3	00	0D0D	80D3			2026
1230	69	1355	80D3	J	69	A0001	80D3		DUPLICATE INTO AVAILABILITY TABLE)	2424
1231	20	0294	1297	D294	20	XXXX2	XXXX2		STORE DYNAMIC LEVEL, RIGHT ENO WORD	1994
1232	69	0D01	1025	J ZQ	69	SETCC	SUB2R		TO RESERVING SUBROUTINE	2444
1233	10	0D03	0709	N BP	10	READC	SUB1D		BY-PASS INVALID CARD	236D
1234	00	0D8D	0D0D	K 85TH	00	0080	D0DD			2614

1235	98	8008	0000		K 8	98	8008	0000				1984
1236	20	0185	0938	0185		20	P0009		P0009	STORE CORE ADDRESS FOR PUNCHING		2218
1237	10	0590	8003			10		8003		MAKE BRINGING ORDER		1962
1238	65	0159	0664		C PRO1	65	R0009			NUMERICAL I		2284
1239	10	0186	0544			10	P0010			FIX C.I. SO THAT I WILL NOT PUNCH)		2300
1240	69	0944	0042			69		SUB19		PROCESS L, IF REGIONAL OR SYMBOLIC		2532
1241	36	0000	1263		N YES	36	0000			WE FOUND A CELL. SHIFT AND COUNT		1708
1242	30	0001	0999			30	0001	A		SHIFT RIGHT FOR CORE ADDRESS		2056
1243	00	0808	0798		J	00	0808			TAGS AND EXIT FOR SUBROUTINE 13		2160
1244	24	0186	1294	0186		24	P0010		P0010	C.I. FOR AVAILABILITY TABLE)		2554
1245	69	1555	0168		O QS	69	A0201	T0001		TO RESTORE BEFORE END OF JOB		2606
1247	69	0197	1100			69	ALOPT			SHOULD WE RESERVE		1712
1248	24	1359	1034	1359	J SENO	24	A0005	A	A0005	DUPLICATE INTO AVAILABILITY TABLE)		2426
1249	20	0177	1334	0177	C LOOP	20	P0001		P0001	STORE LOCATION OF AVAILABILITY WORD		2560
1250	23	2225	2400	2225	K	1250	23	2225	2400	2225	TABLE TO OPTIMIZE SHIFT OPERATIONS	310
1251	07	0607	0600		K	1251	07	0607	0600		TABLE TO OPTIMIZE SHIFT OPERATIONS	312
1252	07	0609	0800		K	1252	07	0609	0800		TABLE TO OPTIMIZE SHIFT OPERATIONS	314
1253	09	0811	1000		K	1253	09	0811	1000		TABLE TO OPTIMIZE SHIFT OPERATIONS	316
1254	11	1013	1200		K	1254	11	1013	1200		TABLE TO OPTIMIZE SHIFT OPERATIONS	318
1255	13	1215	1400		K	1255	13	1215	1400		TABLE TO OPTIMIZE SHIFT OPERATIONS	320
1256	15	1417	1600		K	1256	15	1417	1600		TABLE TO OPTIMIZE SHIFT OPERATIONS	322
1257	17	1619	1800		K	1257	17	1619	1800		TABLE TO OPTIMIZE SHIFT OPERATIONS	324
1258	19	1821	2000		K	1258	19	1821	2000		TABLE TO OPTIMIZE SHIFT OPERATIONS	326
1259	21	2023	2200	2023	K	1259	21	2023	2200	2023	TABLE TO OPTIMIZE SHIFT OPERATIONS	328
1260	65	0515	1211			65	110TH	BT		COUNT OF I TAGS IN LOWER		1950
1261	23	0167	0971	0167		23	XXXX1		XXXX1	STORE LAST 4 DIGITS OF OPTIM		1970
1262	69	1315	1024			69		SUBR8		PROCESS L		2254
1263	20	0294	1247	0294		20	XXXX2		XXXX2	STORE THE COUNT		1710
1264	10	0320	0730			10	501XX			ADVANCE IT		2390
1265	65	0768	0611		N ON8	65		SUB11		USUAL CASE, I BLANK AND NOT 0.		2200
1266	24	1660	0029	1660	J ST	24	G0001	0029	G0001	STORE ADDRESS OF 0001 OF REGION		2492
1267	65	0095	0127		J	65	E	Q0010	Q0010	BRING VALUE OF I INTO LOWER		2338
1268	24	0186	1240	0186		24	P0010		P0010	CONTROL INFORMATION)		2530
1269	23	0536	0704	0536		23	EOUIV	ABS	EQUIV	STORE EQUIVALENT		1494
1270	15	8001	1227			15	8001			L-0 ADDRESS TO LEFT END LOWER)		1794
1271	11	1124	0929			11	27TH			SUBTRACT 2000, TO DETERMINE RANGE		2050
1272	65	8003	0179			65	8003	EXIT2		NEW DYNAMIC LEVEL INTO LOWER		1916
1273	65	0167	0177		J	65	XXXX1	EXITX		EQUIVALENT BACK TO LOWER AND EXIT		1408
1274	90	0978	0573			90		I0002		I, DRUM IS FULL		2196
1275	15	0428	8002			15		8002		MAKE BRINGING ORDER		1840
1276	00	0888	0477		J	00	0888	00009		TAGS AND EXIT FOR USE IN SUBR 11		2142
1277	65	1250	0706		J	65	1250			BRING SHIFT ADDRESS		1824
1278	69	1032	0677			69		IN0EX		IN0EX I		2292
1279	00	0000	8002		K 8002	00	0000	8002				2024
1280	69	1333	0740			69		SUB		TO SUBROUTINE		2450
1281	20	0167	1121	0167		20	XXXX1		XXXX1	STORE ADDRESS		2034
1282	65	0153	1016			65	R0003			ALPHABETICAL I		2460
1283	65	1680	0690		N 00NE	65	ZER0X			CLEAR ACCUMULATOR		2428
1284	15	1344	1249			15	R51	LOOP		INITIAL WORD SHOWING LOCATION OF TABLE		2558
1285	60	8001	1292			60	8001			PUT IT INTO UPPER		2088
1286	65	0158	0614		C PRO0	65	R0008			NUMERICAL 0		2264
1287	45	0640	1042			45	XY			I, 8002-8003 + WRONG PARITY		2008
1288	20	0185	1338	0185		20	P0009	ALL	P0009	STORE I FOR PUNCHING		2296
1289	69	0594	0661		C S0	69		SUBR7		STORE SYMBOL AND EQUIVALENT IN TABLE		2354
1290	30	0004	0602			30	0004					2538
1291	15	0444	8002			15		8002		MAKE SHIFTING ORDER		1726
1292	69	0595	1025			69		SUB2R		RESERVE ADDRESS		2090
1293	00	0809	0998		J	00	0809			TAGS AND EXIT FOR SUBROUTINE 13		2222
1294	60	1348	1284			60	A1			VARIABLE BRINGING ORDER INTO UPPER		2556
1295	00	1499	1499		K C7	00	1499	1499		TO MODIFY IDENTIFICATION AT END OF LI		2612
1297	14	1022	1322			14	21XXX			DIVIDE DYNAMIC LEVEL BY 2		1996
1298	24	1669	1280	1669		24	ORUMT	ORUMT		CHANGE DRUM TAG. DRUM CANNOT BE FULL.		2448
1299	20	0179	0745	0179		20	P0003	P0003		STORE LOCATION OF AVAILABILITY WORD		2564
1300	19	9906	0690		K	1300	19	9906	0690	TABLE TO OPTIMIZE INDEXING OPERATIONS		330
1301	80	0008	0880		K	1301	80	0008	0880	TABLE TO OPTIMIZE INDEXING OPERATIONS		332
1302	80	0106	0680		K	1302	80	0106	0680	TABLE TO OPTIMIZE INDEXING OPERATIONS		334
1303	80	0209	0880		K	1303	80	0209	0880	TABLE TO OPTIMIZE INDEXING OPERATIONS		336
1304	80	0308	0980		K	1304	80	0308	0980	TABLE TO OPTIMIZE INDEXING OPERATIONS		338
1305	99	9908	0880		K	1305	99	9908	0880	TABLE TO OPTIMIZE INDEXING OPERATIONS		340
1306	24	0095	0648	0095		24	OPTIM	OPTIM		STORE OPTIMIZING ADDRESS AND TAGS		1966
1307	11	0610	0415			11	OB			MAKE DRUM AVAILABLE)		974
1308	00	0000	0090		K 99THX	00	0000	0090				2660
1309	30	0004	1220			30	0004			WHAT CELL DID WE RESERVE)		1744
1310	44	0464	0514			44		UM		O, D TAG IS NON-ZERO		1954
1311	15	0564	8002			15		8002		BRINGING ORDER		2042
1312	69	0666	1024			69		SUBR8		PROCESS 0		2268
1313	35	0001	1270			35	0001			L-0 ADDRESS TO LEFT END LOWER)		1792
1314	11	0159	0716			11	R0009			SUBTRACT END OF AREA TO BE RESERVED		2394
1315	44	1321	1324		J	44	NGL			O, RELOCATED L IS EXCESSIVE		2256
1316	35	0008	1040			35	0008			CLEAR ALL EXCEPT LAST CHARACTER)		2504
1317	15	1019	1082		N 0	15	ODIFF	CO		ADD DRUM RELOCATION AMOUNT		2370
1318	24	0185	1290	0185		24	P0009	P0009		STORE LAST 4 DIGITS OF CONSTANT IN I		2536
1319	69	8003	0776		C LR	69	8003			CLEAR DISTRIBUTOR		1552
1320	65	0062	0967			65	OPREG			WHICH ONE DO WE USE)		1886
1321	10	0186	1343		N NGL	10	P0010			FIX C.I. SO THAT L WILL NOT PUNCH)		2260
1322	44	1076	1026			44		EVN		O, DYNAMIC LEVEL IS 000.		1998
1323	65	8002	0731			65	8002			CLEAR UPPER		1430
1324	20	0182	1286	0182		20	P0006	PRO0	P0006	STORE L FOR PUNCHING		2258
1325	69	0179	0632			69	EXITZ			IF BACKWARD L, SUBTRACT TAG-COUNT)		1906
1326	30	0001	1083		N NGO	30	0001			FIX C.I. SO THAT O WILL NOT PUNCH)		2278
1327	65	8002	0985			65	8002	BOTH		ODD. SEPARATE LAST ADDRESS.		1896

1328 00 80D0 00D0	K 83R0	0D 8D0D	0000		2374
1329 70 0151 11D2	N OUT	70 RDD01		WHEN DONE, READ NEXT CARD	2400
1330 65 0159 1066		65 RD009		NUMERICAL I INTO LOWER	2464
1331 00 0001 1236	J	00 D001		ADDEND AND EXIT FOR SUBROUTINE 18.	2216
1332 10 0158 0179	C BL	10 RD008	EXITZ	NUMERICAL 0 INTO UPPER	2468
1333 69 D001 1075	J	69 SETCC	SUB2U	TO UNRESERVING SUBROUTINE	2452
1334 15 D695 1299		15 C3		MODIFY IDENTIFICATION OF WORD	2562
1335 65 D292 1337		65 ORCEQ	MW	DYNAMIC LEVEL OF LAST CORE ADDRESS	2106
1336 20 D184 1238	D184	2D P00D8	PROI	PD008 STORE 0 FOR PUNCHING	2276
1337 15 0545 0335	C MW	15 COUNT	SB10C	CORE OR DRUM. ADD TAG-COUNT TO DYN LV	21D8
1338 6D D003 0D57	C ALL	6D REAOC	SB10A	PREPARE TO PUNCH AND READ NEXT CARD	2304
1339 45 0644 0694		45 ALL		I, IT IS ANOTHER RBD CARD	2406
1340 24 0183 0994	0183	24 P0007		PD007 STORE FIRST 2 DIGITS IN OP POSITION	2542
1341 10 0294 0649		10 XXXX2		WHAT CELL DID WE RESERVE I	1754
1342 0D 0890 0418	J	00 0890	L0001	TAGS AND EXIT FOR USE IN SUBR 11	2100
1343 21 0186 1286	0186	21 P0010	PROO	PD01D FIX C.I. SO THAT L WILL NOT PUNCH I	2262
1344 00 0000 0450	K RS1	00 D0DD	045D	INITIAL IDENTIFICATION OF WORD	26D8
1345 45 0596 0646		45	S0D	IF 0 IS BLANK, WE SHOULD STORE ZERO	2622
1347 16 0536 1092		16 EQUIV		SUBTRACT THE ADDRESS IN QUESTION	2D14
1348 69 1355 0168	P A1	69 A0001	TD001	INITIAL VARIABLE BRINGING ORDER	2600
1349 20 0185 0795	0185	20 P0009		PDD09 STORE LOCATION OF AVAILABILITY WORD	2568
1350 69 0154 1268		69 R0004		CONTROL INFORMATION I	2528
1351 66 0792 0935	N OK	66 IN0	P	MAKE NEGATIVE, BUT DO NOT INTERCHANGE	1882
1352 69 1555 0172	Q Q	69 A0201	T0005	COMPARISON CONSTANT FOR END OF JOB	26D4
1353 90 1008 1058		90	BB1	0, THERE WAS A BLANK BACKWARDS L	1236
1354 60 1355 1088	Q 600	6D A00D1	TA		1764



THE NATIONAL BUREAU OF STANDARDS

The scope of activities of the National Bureau of Standards at its major laboratories in Washington, D.C., and Boulder, Colo., is suggested in the following listing of the divisions and sections engaged in technical work. In general, each section carries out specialized research, development, and engineering in the field indicated by its title. A brief description of the activities, and of the resultant publications, appears on the inside of the front cover.

WASHINGTON, D.C.

ELECTRICITY. Resistance and Reactance. Electrochemistry. Electrical Instruments. Magnetic Measurements. Electrics.

METROLOGY. Photometry and Colorimetry. Refractometry. Photographic Research. Length. Engineering Metrology. Mass and Scale. Volumetry and Densimetry.

HEAT. Temperature Physics. Heat Measurements. Cryogenic Physics. Rheology. Molecular Kinetics. Free Radicals Research. Equation of State. Statistical Physics. Molecular Spectroscopy.

RADIATION PHYSICS. X-Ray. Radioactivity. Radiation Theory. High Energy Radiation. Radiological Equipment. Nucleonic Instrumentation. Neutron Physics.

CHEMISTRY. Surface Chemistry. Organic Chemistry. Analytical Chemistry. Inorganic Chemistry. Electro-Deposition. Molecular Structure and Properties of Gases. Physical Chemistry. Thermochemistry. Spectrochemistry. Pure Substances.

MECHANICS. Sound. Pressure and Vacuum. Fluid Mechanics. Engineering Mechanics. Combustion Controls. ORGANIC AND FIBROUS MATERIALS. Rubber. Textiles. Paper. Leather. Testing and Specifications. Polymer Structure. Plastics. Dental Research.

METALLURGY. Thermal Metallurgy. Chemical Metallurgy. Mechanical Metallurgy. Corrosion. Metal Physics.

MINERAL PRODUCTS. Engineering Ceramics. Glass. Refractories. Enameled Metals. Constitution and Microstructure.

BUILDING RESEARCH. Structural Engineering. Fire Research. Mechanical Systems. Organic Building Materials. Codes and Safety Standards. Heat Transfer. Inorganic Building Materials.

APPLIED MATHEMATICS. Numerical Analysis. Computation. Statistical Engineering. Mathematical Physics.

DATA PROCESSING SYSTEMS. Components and Techniques. Digital Circuitry. Digital Systems. Analog Systems. Applications Engineering.

ATOMIC PHYSICS. Spectroscopy. Radiometry. Mass Spectrometry. Solid State Physics. Electron Physics. Atomic Physics.

INSTRUMENTATION. Engineering Electronics. Electron Devices. Electronic Instrumentation. Mechanical Instruments. Basic Instrumentation.

Office of Weights and Measures.

BOULDER, COLO.

CRYOGENIC ENGINEERING. Cryogenic Equipment. Cryogenic Processes. Properties of Materials. Gas Liquefaction.

IONOSPHERE RESEARCH AND PROPAGATION. Low Frequency and Very Low Frequency Research. Ionosphere Research. Prediction Services. Sun-Earth Relationships. Field Engineering. Radio Warning Services.

RADIO PROPAGATION ENGINEERING. Data Reduction Instrumentation. Radio Noise. Tropospheric Measurements. Tropospheric Analysis. Propagation-Terrain Effects. Radio-Meteorology. Lower Atmosphere Physics.

RADIO STANDARDS. High frequency Electrical Standards. Radio Broadcast Service. Radio and Microwave Materials. Atomic Frequency and Time Standards. Electronic Calibration Center. Millimeter-Wave Research. Microwave Circuit Standards.

RADIO SYSTEMS. High Frequency and Very High Frequency Research. Modulation Research. Antenna Research. Navigation Systems. Space Telecommunications.

UPPER ATMOSPHERE AND SPACE PHYSICS. Upper Atmosphere and Plasma Physics. Ionosphere and Exosphere Scatter. Airglow and Aurora. Ionospheric Radio Astronomy.

